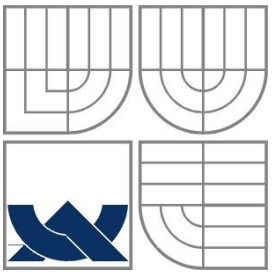
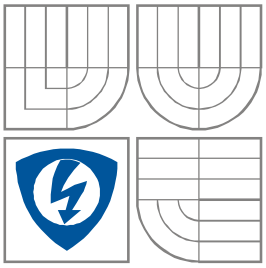


VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

ÚSTAV AUTOMATIZACE A MĚŘICÍ TECHNIKY

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

DEPARTMENT OF CONTROL AND INSTRUMENTATION

MONITOROVACÍ SYSTÉM MOBILNÍCH JEDNOTEK

MONITORING SYSTEM FOR THE MOBILE UNITS

DIPLOMOVÁ PRÁCE

MASTER'S THESIS

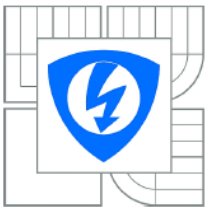
AUTOR PRÁCE — Bc. Pavel Ševčík

AUTHOR

VEDOUCÍ PRÁCE — Ing. Pavel Kučera, Ph.D.

SUPERVISOR

BRNO 2011



VYSOKÉ UČENÍ
TECHNICKÉ V BRNĚ

Fakulta elektrotechniky
a komunikačních technologií

Ústav automatizace a měřicí techniky

Diplomová práce

magisterský navazující studijní obor
Kybernetika, automatizace a měření

Student:	Bc. Pavel Ševčík	***ID:***	70100
Ročník:	2	***Akademický rok:***	2010/2011

NÁZEV TÉMATU:

Monitorovací systém mobilních jednotek

POKYNY PRO VYPRACOVÁNÍ:

Seznamte se stávajícím HW a SW řešením přehledové kamery mobilních jednotek. Navrhněte a realizujte systém pro robustní určení polohy a natočení jednotek s ohledem na různé světelné podmínky. Vytvořte SW vybavení, které implementuje tyto funkce s ohledem na jeho modulárnost, konektivitu a zpracování dat v reálném čase. Pro vytvořený systém proveďte experimenty a definujte chyby měření.

DOPORUČENÁ LITERATURA:

Johnson, M., H.: "Win32 System Programming." Second Edition. Boston, MA, USA. Addison-Wesley, 1997. ISBN 0201634651.
Hlaváč, V., Šonka, M.: "Počítačové vidění." Grada 1992, Praha, ISBN 80-85424-67-3.

Termín zadání:	7.2.2011	***Termín odevzdání:***	23.5.2011

Vedoucí práce: Ing. Pavel Kučera, Ph.D.

prof. Ing. Pavel Jura, CSc.
Předseda oborové rady

UPOZORNĚNÍ:

Abstrakt

Tato práce se zabývá zpracováním obrazu v reálném čase, jeho předzpracováním, segmentací a klasifikací. Na základě klasifikace je určeno natočení a pozice objektů. Cílem práce je vytvořit modulární aplikaci, která bude sledovat mobilní jednotky a určovat jejich natočení a pozici v reálném čase.

Klíčová slova

Zpracování obrazu, počítačové vidění, Houghova transformace, Cannyho hranový detektor, sdílená paměť, mikroprismatické fólie

Abstract

This thesis deals with real-time image processing including preprocessing, segmentation and classification of objects. On the basis of classification is determined rotation and position of objects. The aim of this project is to develop a modular application which will be able to monitor mobile units and determine their rotation and position in real time.

Keywords

Image processing, computer vision, Hough transform, Canny edge detektor, shared memory, microprimatics foils

Bibliografická citace:

Ševčík, P. *Monitorovací systém mobilních jednotek*. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2011. 67s. Vedoucí diplomové práce byl Ing. Pavel Kučera, Ph.D.

Prohlášení

„Prohlašuji, že svou diplomovou práci na téma Monitorovací systém mobilních jednotek jsem vypracoval samostatně pod vedením vedoucího diplomové práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.
Jako autor uvedené diplomové práce dále prohlašuji, že v souvislosti s vytvořením této diplomové práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č. 40/2009 Sb.“

V Brně dne: **23. května 2011** ……

podpis autora

Poděkování

Děkuji vedoucímu diplomové práce Ing. Pavlu Kučerovi, Ph.D. za účinnou metodickou, pedagogickou a odbornou pomoc a další cenné rady při zpracování mé diplomové práce.

Dále bych chtěl poděkovat Ing. Karlu Horákovi, Ph.D. za přínosné rady a konzultace v oblasti počítačového vidění.

V Brně dne: **23. května 2011** ……

podpis autora

Obsah

1 ÚVOD

Cílem této práce je vytvořit počítačový program, který bude monitorovat mobilní jednotky pomocí kamery umístěné nad hřištěm robotického fotbalu. Tento program je navrhován s důrazem na zpracování obrazu v reálném čase a také na modularitu celého systému. Jeho hlavní funkcí je zjišťovat pozice a natočení jednotlivých jednotek na hřišti. V neposlední řadě je cílem této práce vyvinout uživatelské rozhraní, pomocí kterého bude možné monitorovat mobilní jednotky.

První část práce (kapitola 1 – Počítačové zpracování obrazu) se věnuje teorii počítačového vidění a jsou v ní postupně rozebrány základní kroky při počítačovém zpracování obrazu. V této kapitole je nejdříve představena problematika snímání a digitalizace, za kterou následuje obsáhlejší podkapitola Předpracování obrazu. Dále jsou popsány metody segmentace, popisu objektů a klasifikace.

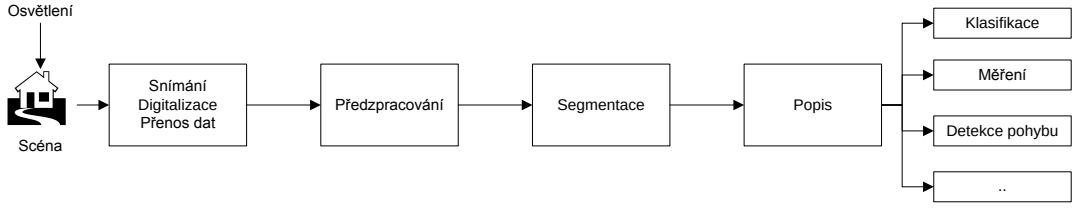
Druhá část práce se věnuje praktické realizaci Monitorovacího systému mobilních jednotek. V této kapitole je nejdříve podrobně popsána scéna, ve které bude zpracování obrazu probíhat. Poté je podrobně popsán kamerový systém a jeho vlastnosti a chyby, které byly v průběhu práce pozorovány. V této části je navrženo označení robotů pomocí mikroprismatických fólií s koutovými odražeči a jsou zde demonstrovány jeho výhody. Poslední část této kapitoly se věnuje návrhu softwarového řešení systému. Protože při návrhu systému je kladem důraz na zpracování v reálném čase, je aplikace vyvíjena pomocí programovacího jazyku C/C++ s využitím knihovny *OpenCV*. Nemalá část práce je věnována výběru metod pro sdílení dat mezi jednotlivými procesy. Dále jsou popisovány jednotlivé metody a algoritmy které byly v aplikaci použity. Nakonec je popsáno grafické uživatelské prostředí a jeho základní funkce.

2 POČÍTAČOVÉ ZPRACOVÁNÍ OBRAZU

Tato kapitola se teoreticky věnuje problematice počítačového vidění. Vědní disciplína počítačového vidění má za cíl teoreticky popsat metody, které vedou alespoň k částečnému pochopení obrazu tak, jak ho chápou lidé. Postup při zpracovávání a rozpoznávání obraz lze rozdělit podle [2] do následujících kroků:

1. Snímání, digitalizace a uložení obrazu.
2. Předzpracování.
3. Segmentace obrazu na objekty.
4. Popis objektů.
5. Klasifikace objektů.

Výše uvedené kroky obecného schématu počítačového vidění jsou postupně rozebrány v následujících podkapitolách. Na Obrázku 2.1 je uvedený rozšířený postup při zpracování obrazu.



Obrázek 2.1: Schéma zpracování obrazu

2.1 Snímání a digitalizace

Metody snímání slouží k tomu, aby byl zkoumaný signál převeden na měřitelnou veličinu. Ve většině případů snímání je převáděna optická veličina na veličinu elektrickou. Měřitelnou elektrickou veličinu je nutné digitalizovat (např. pomocí A/D převodníku). Tato práce se věnuje zpracování obrazu z digitální kamery, proto metody, které nesouvisejí s touto oblastí, budou zmíněny pouze okrajově.

2.1.1 Snímání obrazu

Volbě a nastavení soustavy snímacího zařízení je nutné věnovat zvýšenou pozornost, protože informaci, která je ztracena v důsledku špatného nastavení soustavy snímacího zařízení, již není možné zpětně získat. Příkladem špatného nastavení může být např. nevhodné nastavení optické soustavy před samotným snímačem. Velmi důležité je při snímání také zvolit vhodné nasvícení scény, která má být předmětem zájmu. V současnosti jsou nejvíce rozšířeny dva druhy technologie výroby fotocitlivých snímačů:

- CMOS,
- CCD.

2.1.1.1 CMOS snímač

Podle [5] každý pixel na CMOS čipu obsahuje fotodiodu, která je napojena na tzv. aktivní tranzistor. Každý tranzistor je napojený na čtecí a resetovací obvod. Matice takovýchto detektorů tvoří snímač.
Mezi výhody snímačů s CMOS technologií patří:

- Cena
- Velká rychlost vyčítání
- Menší spotřeba než CCD
- Větší rozsah intenzit
- Možnost mít kameru i procesor na jednom čipu

Nevýhoda CMOS:

- Větší šum oproti CCD

2.1.1.2 CCD snímač

Hlavním prvkem každého čidla CCD snímače je podle [5] je Shottkyho dioda, která převádí světelnou energii na elektrickou. V důsledku tohoto se v připojených kondenzátorech nahromadí energie, která je úměrná dopadajícímu záření. Každý kondenzátor předává svůj náboj sousednímu kondenzátoru. Poslední kondenzátor je napojen na výstupní zesilovač, který převádí elektrický náboj na elektrický signál, jenž je poté převeden do digitální podoby pomocí A/D převodníku. CCD čip funguje jako analogový posuvný registr.
Výhody technologie CCD podle [5]:

- Velká citlivost
- Malý šum

Nevýhody CCD technologie:

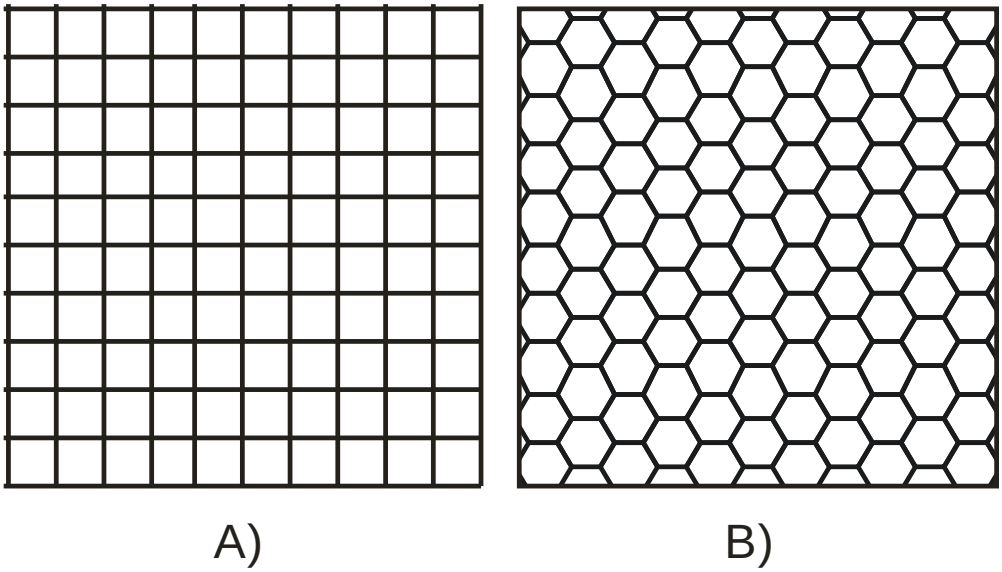
- Vzájemné ovlivňování sousedních pixelů
- Malý rozsah intenzit
- Nemožnost přistupovat k jednotlivým pixelům

2.1.2 Digitalizace

Z fotocitlivých snímačů je získán spojitý elektrický signál, který je nutné převést do digitální podoby. Tento proces se nazývá digitalizace. Digitalizace tedy spočívá ve vzorkování obrazu v matici $M \times N$ bodů a kvantování spojité jasové úrovně do K intervalů. Počet kvantovacích úrovní musí být dostatečně veliký, tak aby byly přesně vyjádřeny i jemné detaily v obraze.

Při vzorkování spojité funkce $f(i,j)$ je nutné určit interval vzorkování. Tento interval (vzdálenost mezi nejbližšími vzorky v obraze) je potřeba volit s ohledem na Shanonnovu větu, která říká, že vzorkovací frekvence se volí jako

dvojnásobek nejvyšší důležité frekvence v obraze. Z této věty vyplývá, že interval vzorkování musí být menší nebo roven polovině nejmenšího detailu ve spojitém obraze. Kromě vzorkovací frekvence je nutné určit plošné uspořádání bodů pro vzorkování. Možná uspořádání jsou zobrazena na Obrázku 2.2., kde jednotlivé uzlové body představují vzorkované body. Na Obrázku 2.2 A) je zobrazena čtvercová vzorkovací mřížka, která je zároveň nejčastěji používanou. Na Obrázku 2.2 B) je zobrazena hexagonální vzorkovací mřížka [2].



Obrázek 2.2: Vzorkovací mřížky: A) čtvercová, B) hexagonální

2.2 Předzpracování obrazu

Cílem předzpracování obrazu je odstranit geometrické zkreslení a potlačit, či zvýraznit některé charakteristické rysy obrazu. Podle [1] je možné rozdělit metody předzpracování na:

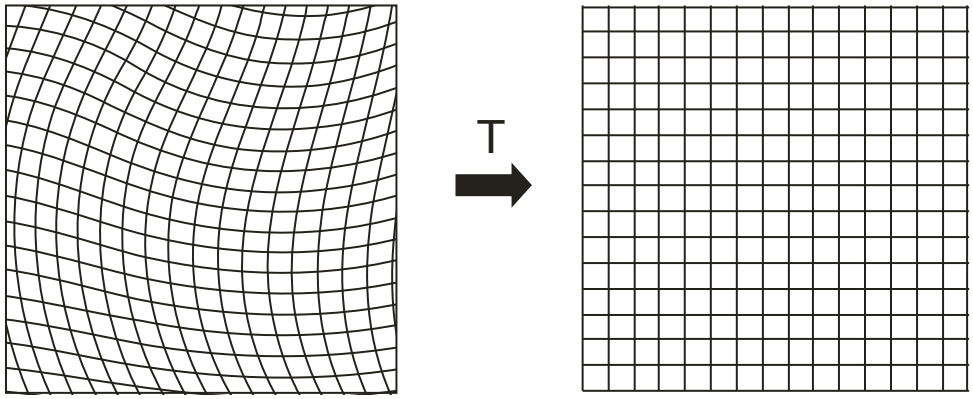
1. Geometrické transformace
2. Bodové jasové transformace
3. Lokální operace předzpracování
4. Restaurace obrazu

2.2.1 Geometrické transformace

Metody geometrické transformace se zabývají tím, jak odstranit geometrické zkreslení z obrazu. Toto zkreslení bývá způsobeno nedokonalostí optické soustavy snímače, špatnou polohou a natočením snímače či nedokonalostí tvaru, který je snímán (např. snímání satelitních snímků Země). V neposlední řadě se geometrické transformace mohou využít realizaci zvětšení, zmenšení, natočení a zkosení obrazu. Obrázek 2.3 ukazuje význam geometrické transformace, kde zkreslený obraz je pomocí transformace *T* převeden do

žádané podoby. Metody geometrické transformace je podle [2] možné rozdělit na:

1. Plošné transformace,
2. Aproximace jasové funkce.



Obrázek 2.3: Význam geometrické transformace

2.2.1.1 Plošné transformace

Úkolem plošné transformace je najít ke každému diskrétnímu bodu zkresleného obrazu souřadnicový ekvivalent v nezkresleném obraze. Tato transformovaná souřadnice výstupního nezkresleného obrazu nemá ve většině případů celočíselný charakter. Transformační vztah souřadnic se podle [2] obvykle aproximuje polynomem m-tého stupně podle vztahu (1):

$$x' = \sum_{r=0}^{m} \sum_{k=0}^{m-r} a_{rk} x^r y^k,$$
$$y' = \sum_{r=0}^{m} \sum_{k=0}^{m-r} b_{rk} x^r y^k, \tag{1}$$

kde x', y' jsou souřadnice bodu ve výstupním obraze a a_{rk}, b_{rk} jsou koeficienty transformace.

Perspektivní zkreslení

Perspektivní zkreslení vzniká v důsledku toho, že optická osa kamery není kolmá na snímanou rovinu. Kamera je tedy nepřesně natočena. Toto zkreslení se dá odstranit pomocí afinní transformace. Podle [6] se afinní zobrazení po zavedení homogenních souřadnic vyjádří pomocí vztahu (2) jako:

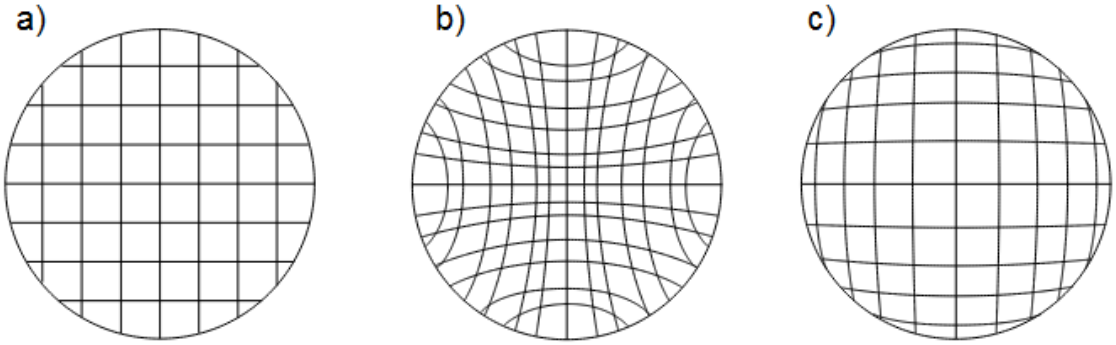
$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} a_1 & a_2 & a_0 \\ b_1 & b_2 & b_0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}, \tag{2}$$

kde a_n a b_n jsou neznámé parametry transformace.

Radiální geometrické zkreslení

Radiální zkreslení je druh vady zobrazení. Tato vada se nejvíce projevuje u širokoúhlých objektivů, ale je možné ji ve větší či menší míře pozorovat u většiny objektivů. Původ radiálního zkreslení je v geometrické nedokonalosti čočky konkrétně v tom, že paprsky z krajních oblastí čočky nesměřují přesně do ohniska. Z tohoto důvodu má obraz různé zvětšení. V literatuře se uvádějí tři základní druhy radiálního zkreslení:

1. Poduškovité (polštářkovité) – pokud jsou vnější části obrazu zvětšeny více než vnitřní (viz Obrázek 2.4 b)
2. Soudkovité – pokud jsou vnější části obraz zvětšeny méně než vnitřní (viz Obrázek 2.4 c).
3. Kombinované – kombinace soudkovitého a poduškovitého zkreslení.



Obrázek 2.4: Radiální zkreslení obrazu: a) nezkreslený obraz, b) poduškovitost, c) soudkovitost, převzato z (Wikipedie, 2011)

Zkreslení se často aproximuje polynomem sudého stupně, pomocí něhož můžeme vadu čočky eliminovat. Podle [3] nezkreslené souřadnice bodů je možné získat pomocí vztahu (3):

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = (1 + k_1 r^2 + k_2 r^4 + k_3 r^6) \cdot \begin{bmatrix} x \\ y \end{bmatrix}, \quad (3)$$

kde x_p a y_p jsou nové souřadnice bodů, x_d a y_d jsou souřadnice bodu ve zkresleném obraze, k jsou neznámé parametry a r je euklidovská vzdálenost od hlavního bodu. Rovnice (3) má tedy pouze tři neznámé koeficienty.

2.2.1.2 Aproximace jasové funkce

Po geometrické transformaci jsou k dispozici vzorky s neceločíselnými souřadnicemi (x', y'). U těchto vzorků je známá hodnota jasové funkce, k zobrazení výsledného obrazu je však nutné dopočítat hodnoty jasu u vzorků s celočíselnými souřadnicemi. Neznámé hodnoty jasu se získávají pomocí interpolačních (aproximačních) metod, které se snaží dopočítat hodnoty jasu

z blízkého okolí bodu [6]. Mezi nejznámější metody interpolace (aproximace) patří:

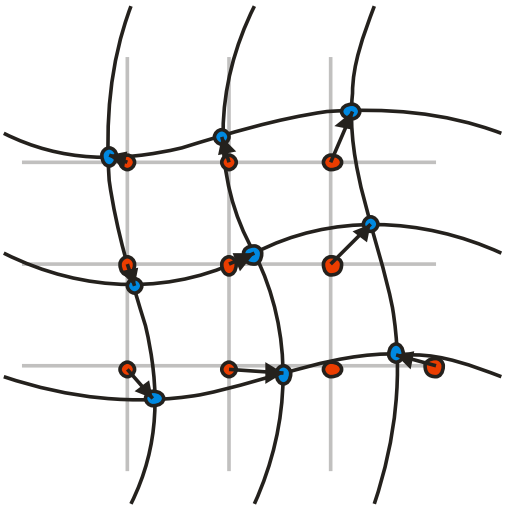
1. Aproximace pomocí nejbližšího souseda.
2. Lineární interpolace.
3. Bikubická interpolace.

Aproximace pomocí nejbližšího souseda

Tato aproximace je podle [6] založena na tom, že přiřadí bodu (x, y) hodnotu jasu nejbližšího souseda g_sv diskrétní mřížce (vztah (4)). Princip této metody je naznačen na Obrázku 2.5.

$$h(x, y) = g_s\big(round(x), round(y)\big), \tag{4}$$

kde $h(x, y)$ je barevná hodnota transformovaného bodu a g_s je původní diskrétní obraz.



Obrázek 2.5: Aproximace pomocí nejbližšího souseda

Mezi výhody této aproximace patří především nízká výpočetní náročnost, proto se používá v real-time aplikacích. Mezi nevýhody je možné zařadit poměrně nekvalitní výstup, vznik nežádoucích artefaktů a neschopnost zajistit hladký přechod barevného tónu[9].

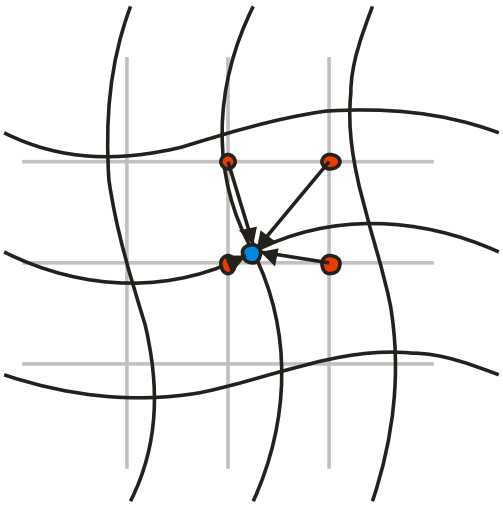
Lineární interpolace

U toho druhu interpolace se využívá vlivu okolních čtyř pixelů. Výsledná jasová hodnota transformovaného bodu je dána lineární kombinací hodnot barev čtyř bodů v jeho okolí. Vliv každého bodu je úměrný blízkosti od transformovaného bodu [6]. Princip výběru okolních bodů je zobrazen na Obrázku 2.6. Výsledná jasová hodnota je popsána pomocí vztahu (5):

$$\begin{aligned} h(x, y) = (1 - a)(1 - b)g_s(l, k) + a(1 - b)g_s(l + 1, k) + \\ + b(1 - a)g_s(l, k + 1) + + abg_s(l + 1, k + 1)\ , \end{aligned} \tag{5}$$

kde

$$l = round(x), k = round(y),$$
$$a = x - l, b = y - k \tag{6}$$



Obrázek 2.6: Lineární interpolace

Ze vztahu (5) je patrné, že výpočet je náročnější, než u metody pomocí nejbližšího souseda. I přes tuto skutečnost se tento algoritmus interpolace řadí mezi rychlejší a často využívané. Jeho nevýhodou je mírné rozmazání obrazu, pokles kontrastu rekonstruované scény a to, že nezajišťuje hladký přechod barevného tónu[9].

Bikubická interpolace

Spočívá v interpolaci obrazové funkce bikubickým polynomem ze 16 bodů v okolí. Výhodou této interpolace je to, že zachovává jemné detaily lépe, než je tomu u předcházejících metod. Nevýhodou této metody je výpočetní náročnost, která ve většině případů předurčuje její použití na aplikace, které požadují kvalitu transformovaného obrazu a nikoli rychlost algoritmu[6] [9].

2.2.2 Bodové jasové transformace

Bodové jasové transformace je možné rozdělit do dvou základních skupin:

1. Jasové korekce.
2. Modifikace jasové stupnice.

2.2.2.1 Jasové korekce

V optických soustavách je obvykle světlo, které prochází optickou soustavou dále od optické osy, zeslabeno. Také jednotlivé fotocitlivé prvky u snímačů nemusejí být stejně citlivé. Pokud jsou tyto poruchy systematické, tak je lze potlačit jasovými korekcemi. Pokud je porušení multiplikativního charakteru, tak je možné pomocí degradační funkce $e(i,j)$ porušení potlačit. Výstupní obraz je poté podle [2] dán pomocí vztahu (7):

$$f(i,j) = e(i,j) \cdot g(i,j), \tag{7}$$

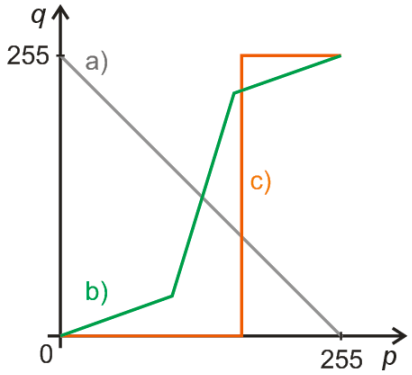
kde $f(i,j)$ je výstupní obraz, $e(i,j)$ je degradační funkce a $g(i,j)$ je původní vstupní obraz.

2.2.2.2 Modifikace jasové stupnice

Tyto transformace nejsou závislé na poloze bodu v obraze. Obecně lze transformaci jasové stupnice popsat vztahem (8):

$$q = T(p), \tag{8}$$

kde q je upravená hodnota jasu, q je původní hodnota jasu a T je transformační funkce. Příklady transformačních funkcí jsou na Obrázku 2.7.



Obrázek 2.7: Transformace jasové funkce: a) negativ, b) zvýšení kontrastu, c) prahování

2.2.3 Lokální operace předzpracování

Metody lokálního předzpracování jsou specifické tím, že výpočet hodnoty jasu bodu ve výstupním obraze je určen na základě okolí odpovídajícího bodu ve vstupním obraze [2]. Operace lokálního předzpracování je možné rozdělit na dvě základní části:

1. Vyhlazování obrazu.
2. Gradientní metody.

Dále je možné rozdělit operace podle funkčního vztahu pro výpočet jasu bodu výsledného obrazu na:

1. Lineární.
2. Nelineární.

Pro lineární operace je výsledný jas v bodě (i,j) podle [2] dán lineární kombinací jasů v okolí O vstupního obrazu g s váhovými koeficienty h:

$$f(i,j) = \sum_{(m,n)} \sum_{\in O} h(i-m, j-n) \cdot g(m,n) \tag{9}$$

Vztah (9) představuje diskrétní konvoluci s konvolučním jádrem h.

2.2.3.1 Vyhlazování obrazu

Žádaným výsledkem vyhlazování obrazu je potlačení vyšších plošných frekvencí obrazové funkce. Jedná se tedy o formu filtrace náhodného šumu v obraze. Negativní vlastností těchto metod je to, že kromě potlačení šumu rozmazávají ostré hrany.

Obyčejné průměrování

Nejjednodušší metodou filtrace je obyčejné průměrování. Výstupní obraz je dán konvolucí vstupního obrazu s konvolučním jádrem definovaným pomocí vztahu (10).

$$h = \frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \tag{10}$$

Základní nevýhodou obyčejného průměrování je to, že značně rozmazává hrany.

Filtrace mediánem

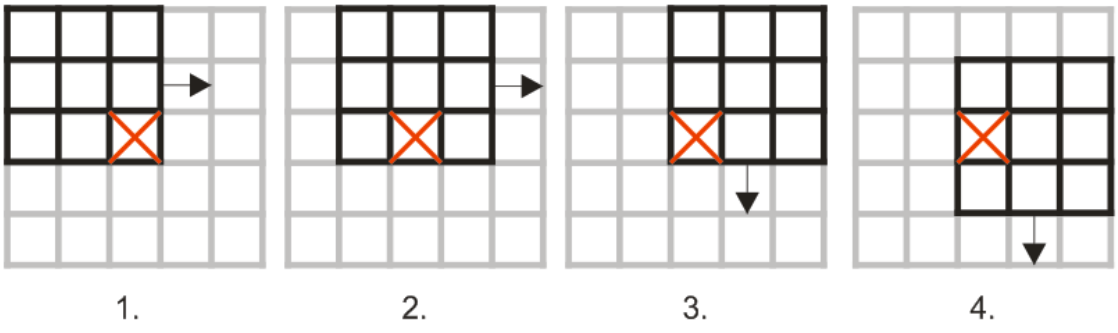
Jedná se o nelineární druh filtrace, která spočívá v tom, že je určen medián hodnot jasu v okolí bodu ve vstupním obraze. Hodnota jasu ve výstupním obraze je pak určena podle vztahu (11):

$$f(i,j) = \widetilde{O_{ij}}\,, \tag{11}$$

kde $\widetilde{O_{ij}}$ je medián jasových hodnot v okolí bodu (i,j) ve vstupním obraze g. Tato metoda redukuje stupeň rozmazání hran a dobře potlačuje impulsní šum. Její nevýhodou je to, že poškozuje úzké čáry. Tuto nevýhodu je možné částečně odstranit použitím jiného než čtvercového druhu okolí.

Filtrace rotující maskou

Tato metoda se snaží najít takové okolí bodu, ve kterém bude nejvyšší homogenita jasu. Příklad principu algoritmu je zobrazen na Obrázku 2.8, kde jsou zobrazeny první čtyři z devíti posunů rotující masky. Při každém kroku se spočítá rozptyl jasu v příslušném okolí. Poté je vybráno takové okolí, kde je rozptyl nejmenší. Výsledná hodnota ve výstupním obraze je dána průměrem hodnot ve zvoleném okolí [2].



Obrázek 2.8: První čtyři kroky metody rotující maskou

Mezi výhody této metody patří to, že nerozmazává hrany a má mírně ostřící charakter.

Filtr s Gaussovým rozložením

U tohoto filtru je definována konvoluční maska pomocí 2D Gaussova rozložení. Gaussovo rozložení je definováno vztahem (12):

$$h(x,y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \tag{11}$$

kde (x,y) jsou souřadnice obrazu a σ je směrodatná odchylka, která udává velikost okolí, ve kterém filtr pracuje [1]. Pomocí vztahu (12) je definovaná konvoluční maska s Gaussovým rozložením se směrodatnou odchylkou $\sigma = 0{,}5$.

$$h = \begin{bmatrix} 0{,}0113 & 0{,}0838 & 0{,}0113 \\ 0{,}0838 & 0{,}6193 & 0{,}0838 \\ 0{,}0113 & 0{,}0838 & 0{,}0113 \end{bmatrix} \tag{12}$$

2.2.3.2 Gradientní operace

Tyto metody vedou ke zvýraznění vyšších prostorových frekvencí. Jsou tedy zvýrazněny hrany, které jsou charakteristické rychlou změnou jasu (vysoký gradient jasové funkce). Negativem těchto metod je to, že kromě hran zvýrazňují i šum v obraze. K nalezení hran v obraze se používají základní dva druhy operátorů

1. Operátory aproximující první derivaci.
2. Operátory aproximující druhou derivaci.

Operátory aproximující první derivaci

Tyto operátory aproximují první parciální derivaci obrazové funkce. Mezi nejjednodušší zástupce této skupiny patří Robertsův operátor, jehož konvoluční masky jsou uvedeny pomocí vztahu (13).

$$h_1 = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}, h_2 = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} \tag{13}$$

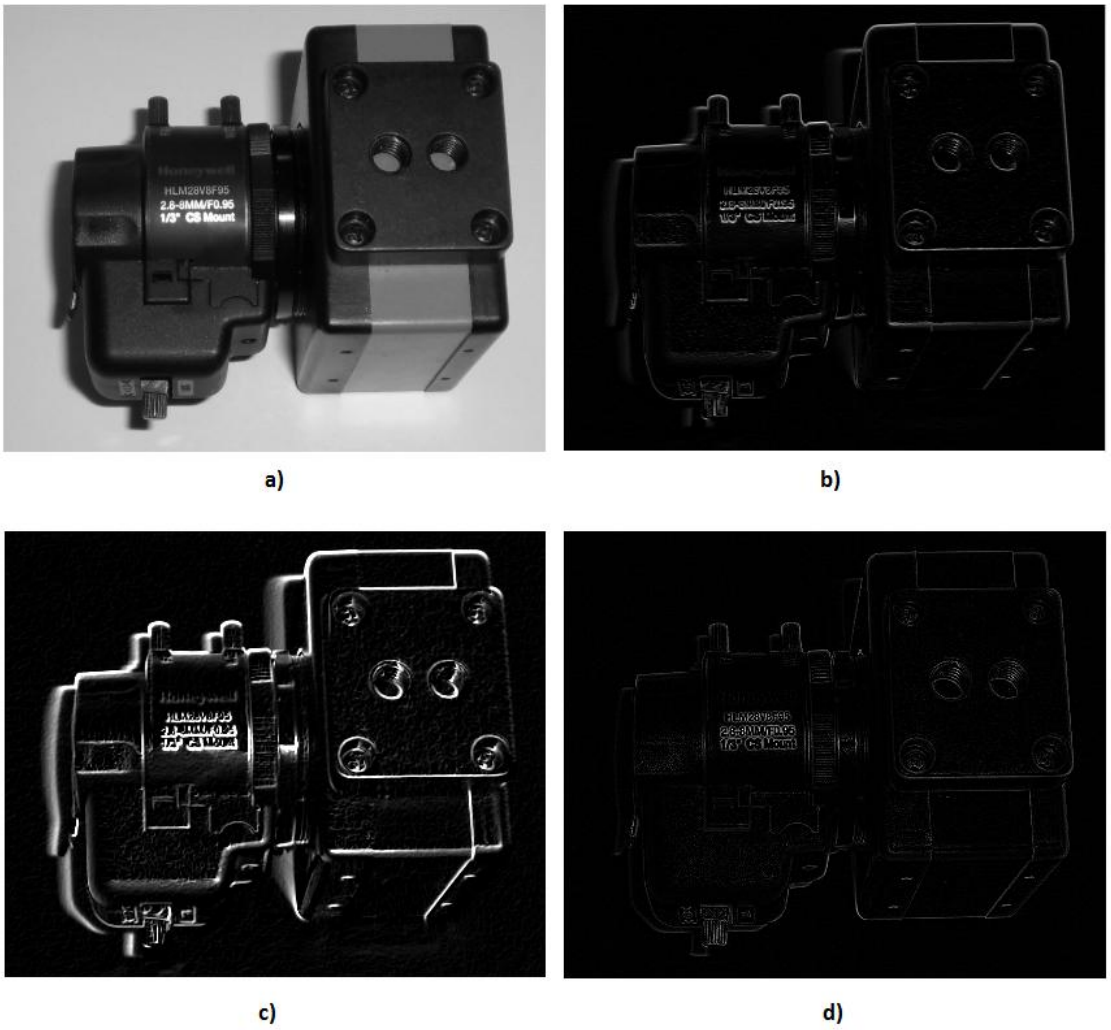
Dalším operátor, který aproximuje první derivaci, je Sobelův operátor, jenž je stejně jako Robertsův operátor, směrově závislý. Konvoluční maska Sobelova operátoru je uvedena pomocí vztahu (14).

$$h_1 = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}, h_2 = \begin{bmatrix} 0 & 1 & 2 \\ -1 & 0 & 1 \\ -2 & -1 & 0 \end{bmatrix} \tag{14}$$

Operátory aproximující druhou derivaci

Mezi tyto operátory patří Laplacelův gradientní operátor, který je necitlivý na natočení a udává pouze velikost hrany, nikoliv orientaci. Konvoluční masky Laplaceova gradientního operátoru jsou uvedeny pomoci vztahu (15).

$$h_1 = \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}, h_2 = \begin{bmatrix} 1 & 1 & 1 \\ 1 & -8 & 1 \\ 1 & 1 & 1 \end{bmatrix} \quad (15)$$



Obrázek 2.9: Ukázka použití hranových detektorů: a) originální šedotónový obraz, b) obraz hran pomocí Robertsova operátoru, c) obraz hran získaný pomocí Sobelova operátoru, d) obraz hran získaný pomocí Laplaceova operátoru

Cannyho hranový detektor

Jedná se o algoritmus vyvinutý Johnem F. Cannym. Tento algoritmus byl publikován již v roce 1986 a i po dvaceti pěti letech od svého představení je jedním z nejvyužívanějších hranových detektorů. Podle [8] je možné algoritmus ve zjednodušené formě rozdělit do následujících kroků:

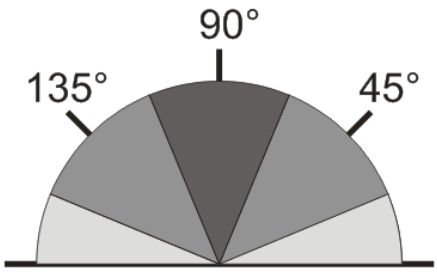
1. Filtrace šumu pomocí Gaussiánu.
2. Detekce hran některým z kompasových detektorů + výpočet absolutní hodnoty gradientu v obrazu podle vztahu (16):

$$|\nabla g(i,j)| = \sqrt{g_x(i,j)^2 + g_y(i,j)^2} \quad (16)$$

3. Výpočet směru gradientu podle vztahu (17):

$$\Theta(i,j) = arctan\left(\frac{g_y(i,j)}{g_x(i,j)}\right) \tag{17}$$

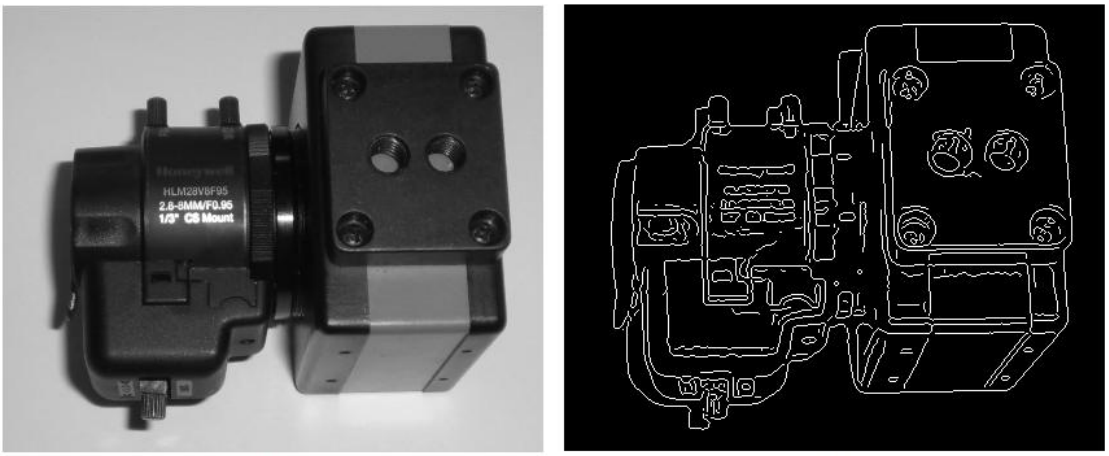
4. Určení směru hrany v každém bodu a normalizace do jednoho ze čtyř směrů (viz Obrázek 2.XX).



Obrázek 2.10: Čtyři oblasti směrů hran

5. Postup po hraně (v jejím směru) a potlačení všech pixelů, které nejsou vyhodnoceny jako hranové.
6. Prahování s hysterezí.

Na Obrázku 2.11 a) je zobrazen původní obraz a na Obrázku 2.11 b) je zobrazen binární obraz získaný pomocí Cannyho hranového detektoru.



Obrázek 2.11: Ukázka Cannyho hranového detektoru: a) originální šedotónový obraz, b) hranová reprezentace získaná pomocí Cannyho hranového detektoru

2.3 Segmentace

Cílem segmentace je podle [1] rozčlenit obraz na části, které souvisejí s předměty či oblastmi reálného světa. Obraz je tedy rozčleněn na pozadí a objekty. Díky segmentaci je pak v dalších krocích zpracování obrazu možné analyzovat jednotlivé objekty, což vede k větší názornosti a úspornosti výpočetních prostředků. Podle [1] je možné metody segmentace rozdělit na:

1. Segmentace prahováním.
2. Segmentace na základě detekce hran.
3. Segmentace narůstáním oblastí.
4. Segmentace srovnáním se vzorem.

2.3.1 Segmentace prahováním

Prahování je nejstarší a nejjednodušší segmentační metodou. Díky své jednoduchosti je výpočetně nenáročná, a proto je stále v některých případech používána. Metoda vychází z předpokladu, že objekty mají konstantní odrazivost či pohltivost povrchu. Zároveň je předpokládáno, že tyto parametry objektů jsou různé od pozadí. Za těchto předpokladů lze oddělit objekty od pozadí pomocí prahování. Prahování je definováno pomocí vztahu (18):

$$g(i,j) = \begin{cases} 1 \text{ pro } f(i,j) \geq T \\ 0 \text{ pro } f(i,j) < T \end{cases} \tag{18}$$

Vztah (18) definuje jednoprahové globální binární prahování, což je nejjednodušší forma prahování. Zásadním parametrem je volba prahu T, který určuje, do jaké množiny bude daný bod spadat.
Obecně existuje jen malé množství případů, kdy lze použít binární prahování. Případem, kdy tento druh prahování selhává, může být nerovnoměrné nasvícená scéna nebo scéna s větším počtem objektů s různými povrchovými vlastnostmi. Problém více objektů s různými povrchovými vlastnostmi řeší prahování s více prahy, které je možné definovat podle vztahu (19):

$$g(i,j) = \begin{cases} 1 \text{ pro } f(i,j) \in D_1 \\ 2 \text{ pro } f(i,j) \in D_2 \\ \vdots \\ n \text{ pro } f(i,j) \in D_n \\ 0 \text{ jinak} \end{cases}, \tag{19}$$

kde je D_n je podmnožinou jasových úrovní.

Problém nerovnoměrného nasvícení scény může vyřešit prahování s lokálním prahem. Tato modifikuje klasické binární prahování v tom, že globální práh $T(f)$ je nahrazen lokálním prahem $T\big(f(i,j)\big)$ získaným z okolí bodu (i,j). Lokální prahování je pak definováno pomocí vztahu (20).

$$g(i,j) = \begin{cases} 1 \text{ pro } f(i,j) \geq T\big(f(i,j)\big) \\ 0 \text{ pro } f(i,j) < T\big(f(i,j)\big) \end{cases} \tag{20}$$

Další variantou je tzv. prahování s hysterezí. Tato metoda používá dvou prahů (horní práh T_h a dolní T_d). Prahování se pak řídí podle následujících pravidel:

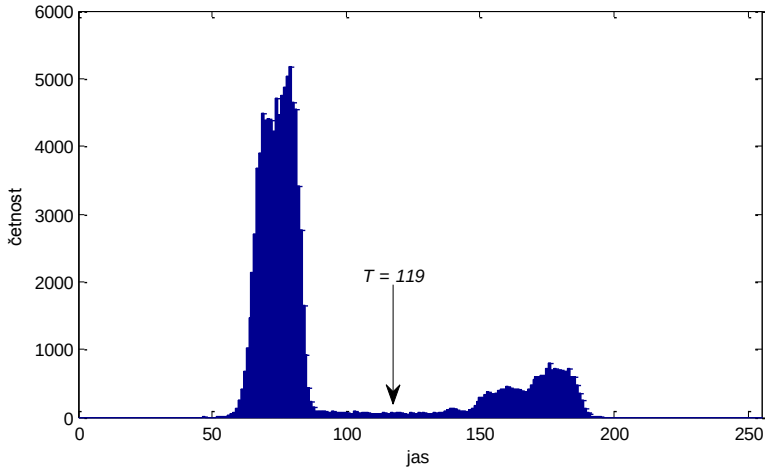
1. Pokud $f(i,j) \geq T_h$, pak bod (i,j) je chápán jako objekt.

2. Pokud $f(i,j) < T_d$, pak je bod (i,j) chápán jako pozadí.
3. Pokud $f(i,j) < T_h \wedge f(i,j) \geq T_d$ a bod (i,j) sousedí s bodem, pro který platí, že $f(i,j) \geq T_h$, pak je tento bod považován za objekt. Při nesplnění těchto podmínek je bod považován za pozadí.

2.3.1.1 Metody určení prahu

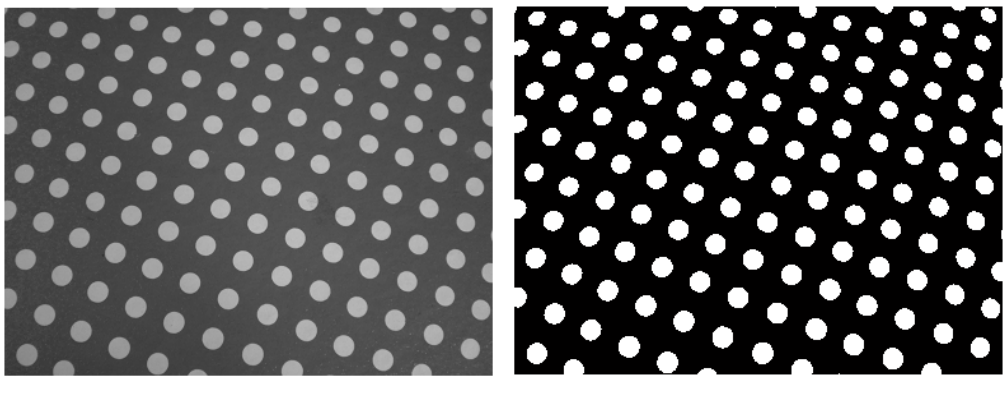
Určit správný práh je nejdůležitější část prahování. Při apriorní znalosti scény a osvětlení je možné práh určit pomocí procentního prahování, které vychází ze znalosti procentuálního zastoupení jasových hodnot objektů a pozadí. Hodnota prahu se poté určí pomocí histogramu daného obrazu.

Další metoda vychází z analýzy tvaru histogramu. Příklad histogramu reálného obrazu je na Obrázku 2.12, kde je možné pozorovat dvě výraznější lokální maxima. První z těchto maxim představuje pozadí a druhé objekty. Za tohoto předpokladu je nejoptimálnější zvolit práh mezi těmito maximy [2].



Obrázek 2.12: Nalezení prahu pomocí tvaru histogramu

Na Obrázku 2.13 a) je obraz, z něhož byl udělán výše zmíněný histogram. Na Obrázku 2.13 b) je binární obraz, který vznikl prahováním s prahem $T = 119$.



Obrázek 2.13: Ukázka prahování s prahem určeným podle tvaru histogramu: a) originální šedotónový obraz, b) vyprahovaný obraz s prahem T = 119

2.3.2Segmentace na základě hran

Tyto metody používají k segmentaci obraz, jenž je vytvořen pomocí některého z hranových operátorů, nebo apriorní informací o objektech. Podle [1] je možné metody na základě segmentace hran rozdělit:

1. Sledování hranice
2. Heuristické sledování hranice
3. Určování hranice s využitím znalosti o její poloze
4. Vyhledávání hranic pomocí Houghovy transformace

2.3.2.1 Vyhledávání hranic pomocí Houghovy transformace

K nalezení hranic objektů v hranové reprezentaci obrazu lze použít Houghovu transformaci. Tuto transformaci je vhodné použít, jsou-li k dispozici apriorní znalosti o daném objektu. Dále je nutné, aby daný objekt bylo možné analyticky nebo parametricky popsat.

Houghova transformace

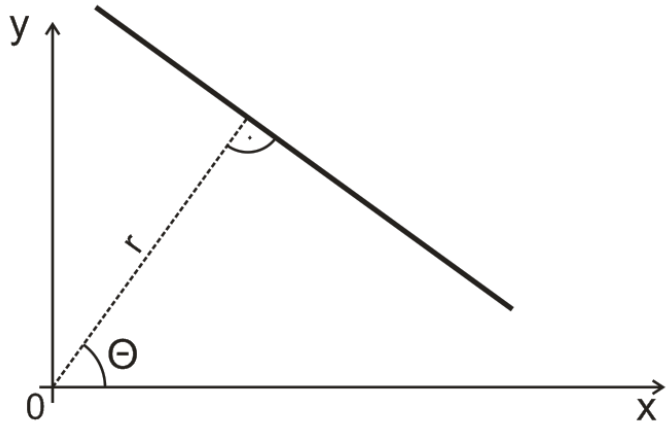
Jedná se o metodu, která umožňuje nalézt v obraze objekt, který lze analyticky popsat. Pomocí této metody lze v obraze nalézt parametry jednoduchých tvarů, jako jsou např.: elipsa, kružnice, přímka či trojúhelník. Pro obecné tvary, které nejsou jednoduše analyticky popsatelné, existuje zobecněná Houghova transformace [1]. Algoritmus diskrétní Houghovy transformace může být popsán pomocí následujících kroků [7]:

1. Vstupem do Houghovy transformace je binární hranová reprezentace obrazu.
2. Prostor parametrů (hledaného objektu či tvaru) je diskretizován.
3. Pro každý uzlový bod v prostoru parametrů je zřízen čítač.
4. Pro každý hranový pixel jsou inkrementovány ty čítače, jejichž souřadnice (parametry křivky) vyhovují rovnici křivky (aproximativně)
5. Po zpracování všech pixelů hranové reprezentace je řešením vektor parametrů, jehož čítač bude mít nejvyšší hodnotu. Je možné stanovit i jiná výstupní pravidla.

Pro detekci přímek se podle [1] příliš nepoužívá směrnicový tvar přímky (21), protože interval možných hodnot parametru k je množina všech reálných čísel. Z tohoto důvodu se využívá tzv. normálový tvar přímky, který je vyjádřen pomocí vztahu (22). Na Obrázku 2.14 jsou vyznačeny jednotlivé parametry normálového tvaru přímky.

$$y = kx + q \tag{21}$$

$$x \cos \Theta + y \sin \Theta = r \tag{22}$$



Obrázek 2.14: Normálový tvar přímky

Pro hledání kružnic se pracuje s vztahem (23) nebo parametrickým tvarem rovnice daným vztahem (24). Tyto vztahy obsahují tři neznámé parametry, proto i Houghův prostor parametrů bude mít tři rozměry. S rostoucím počtem parametrů roste i výpočetní náročnost algoritmu.

$$(x - a)^2 - (y - b)^2 = r \tag{23}$$

$$\begin{aligned} x &= a + r \cos \alpha \\ y &= b + r \sin \alpha \end{aligned} \tag{24}$$

Hlavní výhoda Houghovy transformace je to, že není příliš citlivá na zašuměný signál a na porušení hranic objektu. Mezi nevýhody patří výpočetní náročnost a nižší přesnost[1].

2.3.3Segmentace narůstáním oblastí

Tyto metody se uplatňují v obrazech zatížených šumem, na kterých mnohé ostatní metody selhávají. Základní myšlenkou těchto metod je rozčlenit obraz do maximálních souvislých oblastí tak, aby byly z hlediska zvoleného způsobu homogenní [2]. Kritéria homogenity mohou být založená na hodnotách jasových, texturních či barvených.

2.3.3.1 Spojování oblastí

Jedná se o metodu, při které je vstupní obraz rozdělen na elementární oblasti (2x2 pixelu, 4x4 pixelu, …) a poté je podle [2] podroben následujícímu algoritmu:

1. Definice počátečního rozdělení obrazu do velkého množství malých oblastí.

2. Definice kritéria spojování dvou sousedních oblastí.
3. Sousední oblasti splňující dané kritérium se spojí. Pokud nelze nalézt žádné dvě oblasti, které by splňovaly dané kritérium, tak je algoritmus ukončen.

2.3.3.2 Štěpení oblastí

Jedná se o principiálně opačný postup k segmentaci, než u spojování oblastí. Algoritmus postupuje tak, že je kritériu homogenity nejdříve vystavena celá oblast obrazu. Pokud neodpovídá kritériu homogenity, tak se dělí do menších oblastí, které podstupují stejná kritéria[1].

2.3.3.3 Segmentace srovnání se vzorem

Jedná se o metody, které k segmentaci používají srovnání se vzorem. Tyto metody předpokládají apriorní znalost o segmentovaných objektech. V reálných obrazech lze docílit absolutní shody se vzorem pouze v minimu případů, proto se u těchto metod zavádí kriteriální funkce, která kvantifikuje míru shody objektu se vzorem[1].

2.4 Popis objektů

Jako předposlední část zpracování obrazu je popis objektů. Úkolem této části je získat příznaky vysegmentovaných objektů. Příznaky by měly být voleny tak, aby co nejlépe popisovaly daný objekt a zároveň aby jej odlišovaly od ostatních druhů objektů. Výstupem popisu je tzv. příznakový vektor, který zpracovávají a vyhodnocují algoritmy klasifikace. Příznakový vektor je definován pomocí vztahu (25):

$$X = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} \tag{25}$$

Příznaky lze podle [1] rozdělit podle mnoha hledisek. Mezi základní tři patří základní doména popisované vlastnosti, oblast výpočtu a oblast popisu.
Podle domény popisované vlastnosti je možné deskriptory[1] rozdělit na:

1. **Fotometrické** – popisují optické vlastnosti objektu.
2. **Radiometrické** – popisují geometrické vlastnosti objektu

Podle oblasti výpočtu lze deskriptory dělit:

[1] Deskriptor = příznak, klíčová vlastnost.

1. **Deskriptory založené na regionech** – založené na znalosti jasových hodnot jednotlivých pixelů.
2. **Deskriptory založené na hranicích** – založené na znalosti hranice objektu.

Podle oblasti popisu:

1. **Globální deskriptory obrazu** – popisuje obraz jako celek.
2. **Globální deskriptory objektu** – daný příznak je přiřazen popisovanému objektu.
3. **Lokální deskriptory objektu** – daný příznak je počítán pro určitou část objektu.

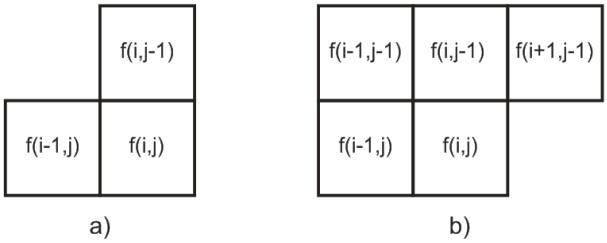
2.4.1 Identifikace oblastí

Výsledkem segmentačních metod je obraz, který se skládá z objektů a pozadí. Aby bylo možné jednotlivé objekty popsat, je nutné jim přiřadit unikátní index. Tento proces se nazývá identifikace oblastí.

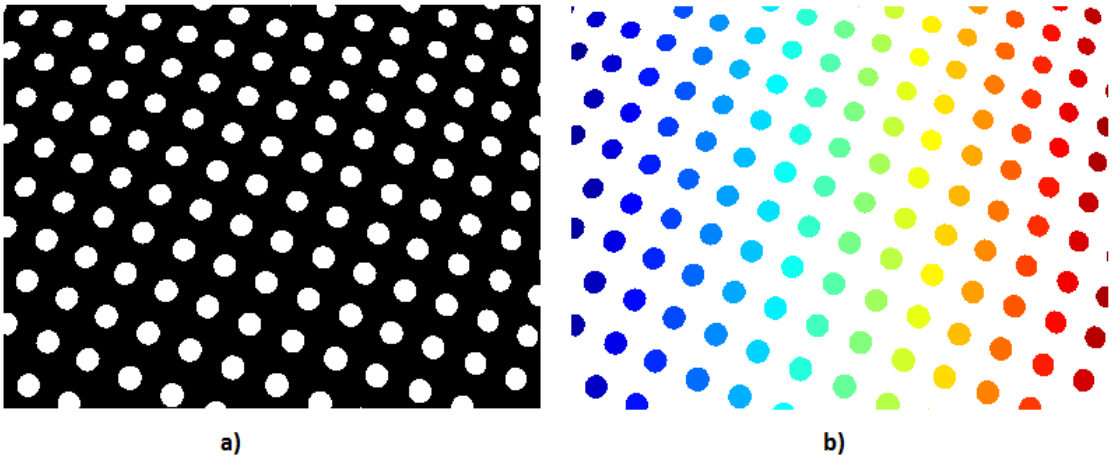
2.4.1.1 Barvení oblastí

Barvení oblastí je dvouprůchodový algoritmus, který umožňuje identifikovat jednotlivé oblasti ve vysegmentovaném obraze. Algoritmus se skládá z následujících kroků:

1. Obraz je postupně procházen po řádcích a každému nenulovému obrazovému elementu podle hodnoty všech jeho již obarvených sousedů je přiřazena určitá hodnota:
   1.1. Jsou – li všechny okolní (viz Obrázek 2.15) body nulové, přiřadíme bodu dosud nepřidělenou barvu (index).
   1.2. Pokud je jeden nebo více nenulových bodů se stejnou barvou (indexem), tak tomuto bodu přiřadíme stejnou hodnotu.
   1.3. Pokud je jeden nebo více bodů nenulových a mají různou barvu, přiřadíme tomuto bodu jednu z okolních barev a všechny barvy z okolí zaznamenáme do tzv. *„tabulky ekvivalence barev“*.
2. Procházíme znovu celý obraz po řádcích a podle tabulky ekvivalentních barev přebarvuje jednotlivé kolizní barvy.



Obrázek 2.15: Masky okolních bodů pro barvení oblastí: a) maska pro 4-sousedství, b) maska pro 8-sousedství



Obrázek 2.16: Příklad obarvených oblastí: a) vysegmentovaný obraz, b) obraz vzniklý po použití algoritmu barvení oblastí

2.4.2 Radiometrické příznaky založené na regionech

Následující podkapitola popisuje vybrané radiometrické příznaky. Tyto příznaky popisují metrické vlastnosti pixelů, které jsou vypočteny z plošného rozložení pixelů objektu [1].

2.4.2.1 Velikost

Jedná se o příznak, jehož hodnota odpovídá počtu pixelů, které tvoří plochu objektu.

2.4.2.2 Obvod

Počet hraničních bodů objektu. Velikost příznaku závisí na tom, jestli je použito k výpočtu 4-okolí nebo 8-okolí.

2.4.2.3 Nekompaktnost

Jedná se o míru podobnosti oblasti k ideálnímu kruhu. Nekompaktnost $x_{nek.}$ je definována pomocí vztahu (26).

$$x_{nek.} = \frac{obvod^2}{velikost} \tag{26}$$

Pokud nekompaktnost je podělena hodnotou 4π, tak je získána normovaná nekompaktnost, přičemž hodnoty 1 nabývá normovaná nekompaktnost, pokud zkoumaný objekt je ideální kruh. Čím je hodnota vyšší, tím je objekt členitější.

2.4.2.4 Konvexnost

Udává míru podobnosti objektu k jeho konvexní schránce. Obrys konvexní schránky je možné si představit tak, že na zkoumaný objekt je navléknuta gumička, která udává konvexní obal předmětu. Konvexnost je dána pomocí vzorce (27).

$$x_{konv.} = \frac{velikost}{plocha\_konvexního\_obalu} \tag{27}$$

Hodnota konvexnosti se pohybuje v intervalu $\langle 0,1 \rangle$, přičemž hodnoty 1 nabývá, pokud je daný objekt „vypouklý“.

2.4.2.5 Minimální poloměr kružnice opsané

Hodnotu tohoto příznaku určuje poloměr minimální kružnice opsané. Příklad je zobrazen na Obrázku 2.17

Obrázek 2.17: Minimální poloměr kružnice opsané

2.4.2.6 Výstřednost

Je dána poměrem délek nejdelších na sebe kolmých tětiv. Jiný způsob určení výstřednosti je pomocí centrálních momentů oblasti. Výstřednost je definována pomocí vztahu (28):

$$x_{výstř.} = \frac{(\mu_{02} + \mu_{20})^2 + 4\mu_{11}}{\mu_{00}} \tag{28}$$

kde μ_{mn} je centrální moment řádu $(m + n)$.

2.4.2.7 Pravoúhlost

Tento příznak je dán maximálním poměrem velikosti a plochy opsaného obdélníku. Minimální plocha opsaného obdélníku je získávána postupným natáčením vysegmentovaného objektu. Pravoúhlost je pak dána pomocí vztahu (29).

$$x_{pravoúhl.} = \frac{velikost}{plocha\_opsaného\_obdélníku} \tag{29}$$

2.4.2.8 Eulerovo číslo

Jeho hodnota je určena jako počet souvislých oblastí objektu mínus počet děr objektu. Na Obrázku 2.18 je zobrazen objekt s jednou souvislou oblastí a dvěma dírami. Výsledná hodnota Eulerova čísla je tedy 1 – 2 = -1.

Obrázek 2.18: Objekt s Eulerovým číslem -1

2.4.3 Geometrické momenty a momentové charakteristiky

Podle [2] momentový popis oblastí interpretuje normalizovanou jasovou funkci obrazu jako hustotu pravděpodobnosti dvojrozměrné náhodné veličiny. Vlastnosti této veličiny lze vyjádřit pomocí statických charakteristik – momentů.

2.4.3.1 Obecné momenty

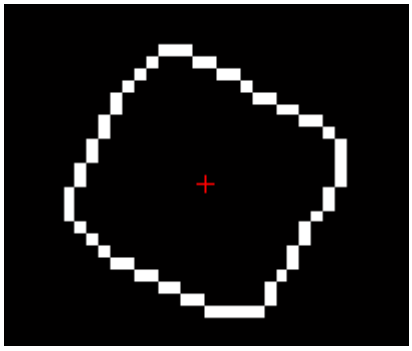
Obecné momenty řádu $(p + q)$ jsou v digitálních obrazech definovány pomocí vztahu (30). Tyto momenty nejsou invariantní vůči změně měřítka, posunutí ani vůči natočení.

$$m_{pq} = \sum_{i=-\infty}^{\infty} \sum_{j=-\infty}^{\infty} i^p j^q f(i,j) \tag{30}$$

Pomocí obecných momentů lze vypočítat souřadnice geometrického těžiště podle vzorce (31):

$$x_t = \frac{m_{10}}{m_{00}}$$
$$y_t = \frac{m_{01}}{m_{00}} \tag{31}$$

kde x_t, y_t jsou souřadnice těžiště a m_{pq} je obecný moment řádu $(p+q)$. Obecný moment prvního řádu odpovídá střední hodnotě v dané souřadnici. Na Obrázku 2.19 je červeným křížkem označeno těžiště daného objektu.



Obrázek 2.19: Příklad těžiště objektu

2.4.3.2 Centrální momenty

Pro výpočet centrálních momentů je nutné znát polohu geometrického těžiště. Centrální moment řádu $(p+q)$ je dán vztahem (32):

$$\mu_{pq} = \sum_i \sum_j (x - x_t)^p \cdot (y - y_t)^q \cdot f(i,j) \tag{32}$$

Centrální momenty jsou invariantní vůči posunu, ale nejsou invariantní vůči otočení a změně měřítka. Centrální momenty μ_{02}a μ_{20} představují rozptyly v daných osách.

2.4.3.3 Normované centrální momenty

Normované centrální momenty jsou invariantní vůči posunutí i vůči změně měřítka. Vzorec pro výpočet normovaného centrálního momentu je vyjádřen pomocí vztahu (33):

$$\vartheta_{pq} = \frac{\mu_{pq}}{(\mu_{00})^\gamma}$$
$$\gamma = INT\left(\frac{p+q}{2}\right) + 1 \tag{34}$$

kde $INT(x)$ značí celočíselnou část čísla x.

2.4.3.4 Momentové charakteristiky (invarianty)

Jedná se o nejstarší sadu sedmi momentových invariantů, které jsou invariantní vůči translaci, rotaci a změně měřítka. Vzorce (35) až (41) popisují jednotlivé momentové invarianty[2].

$$\varphi_1 = \vartheta_{20} + \vartheta_{02} \tag{35}$$

$$\varphi_2 = (\vartheta_{20} - \vartheta_{02})^2 + 4\vartheta_{11}^2 \tag{36}$$

$$\varphi_3 = (\vartheta_{30} - 3\vartheta_{12})^2 + (3\vartheta_{21} - \vartheta_{03})^2 \tag{37}$$

$$\varphi_4 = (\vartheta_{30} + \vartheta_{12})^2 + (\vartheta_{21} + \vartheta_{03})^2 \tag{38}$$

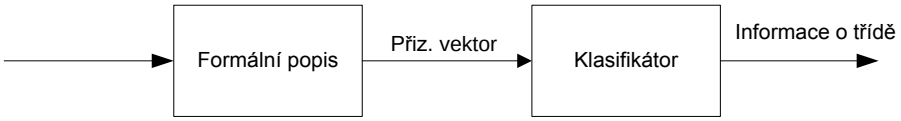
$$\varphi_5 = (\vartheta_{30} - 3\vartheta_{12})(\vartheta_{30} + \vartheta_{12})[(\vartheta_{30} + \vartheta_{12})^2 - 3(\vartheta_{21} + \vartheta_{03})] + \\ +(3\vartheta_{21} - \vartheta_{03})(\vartheta_{21} + \vartheta_{03})[3(\vartheta_{30} + \vartheta_{12})^2 - (\vartheta_{21} + \vartheta_{03})^2] \tag{39}$$

$$\varphi_6 = (\vartheta_{20} - \vartheta_{02})[(\vartheta_{30} + \vartheta_{12})^2 - (\vartheta_{21} + \vartheta_{03})^2] + \\ +4\vartheta_{11}(\vartheta_{30} + \vartheta_{12})(\vartheta_{21} + \vartheta_{03}) \tag{40}$$

$$\varphi_7 = (3\vartheta_{21} - \vartheta_{03})(\vartheta_{30} + \vartheta_{12})[(\vartheta_{30} + \vartheta_{12})^2 - 3(\vartheta_{21} + \vartheta_{03})^2] - \\ -(\vartheta_{30} - 3\vartheta_{12})(\vartheta_{30} - 3\vartheta_{12})[(3\vartheta_{30} + \vartheta_{12})^2 - (\vartheta_{21} + \vartheta_{03})^2] \tag{41}$$

2.5 Klasifikace objektů

Klasifikace objektů je poslední částí zpracování obrazu. Vstupem do klasifikace je příznakový vektor a výstupem je informace o třídě. Jednotlivé třídy představují podmnožiny příznakového prostoru, které mají určité společné rysy. Úkolem klasifikace je zařazení jednotlivých objektů do tříd. Stroj, který tento úkon vykonává, se nazývá klasifikátor.



Obrázek 2.20: Pozice klasifikátoru v řetězci zpracovávání obrazu

Druhů klasifikátorů existuje veliké množství. Výběr konkrétního klasifikátoru záleží např. na počtu tréninkových dat (počet různých příznakových vektorů, které reprezentují jednotlivé objekty za různých podmínek), složitosti řešené úlohy a požadavcích na rychlost klasifikace.

Tato diplomová práce se nevěnuje strojovému učení, a proto zde budou jednotlivé druhy klasifikátorů popisovány pouze okrajově. Mezi vybrané klasifikátory lze zařadit:

1. Rozhodovací stromy
2. Neuronové sítě

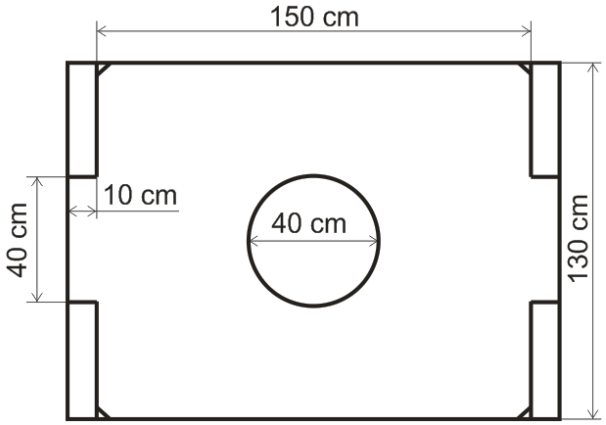
3. Učení založené na instancích – IBL

3 PRAKTICKÁ REALIZACE

Tato kapitola se zabývá praktickou realizací monitorovacího systému mobilních jednotek. Nejdříve je zde zmíněn současný stav a podmínky, v kterých se systém bude nalézat, a poté je popsáno výsledné softwarové řešení s prezentací chyb natočení.

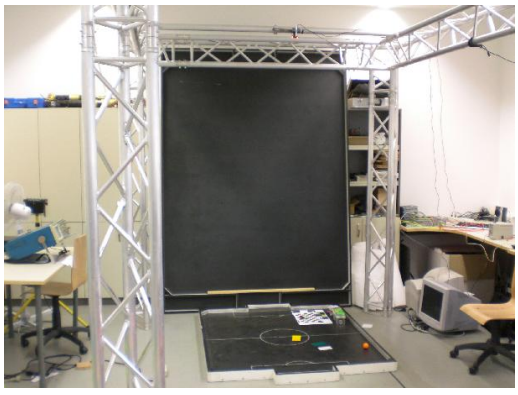
3.1 Vlastnosti scény

Současná laboratoř, která obsahuje prostředky pro monitorovací systém mobilních jednotek, se skládá ze dvou hřišť a kovového rámu, na který je možné umístit kameru. Rozměry menšího hřiště, které bude využito, jsou zobrazeny na Obrázku 3.1.

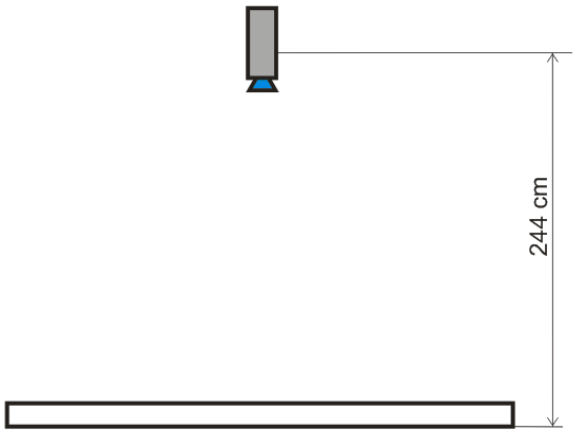


Obrázek 3.1: Rozměry hřiště

Kovová konstrukce, která je nad hřištěm, obsahuje držák, pomocí něhož lze uchytit kameru. Držák je ve vzdálenosti 244 cm od hřiště. Umístění kamery je možné popsat pomocí Obrázku 3.3, na kterém je zakótována vzdálenost kolmá vzdálenost od hřiště k metrickému zavitu pro šroub k upevnění kamery. Rozměry jednotlivých robotů jsou 77,5 × 77,5 × cca 75 mm. Na Obrázku 3.2 je fotografie scény v laboratoři.



Obrázek 3.2: Fotografie scény v laboratoři



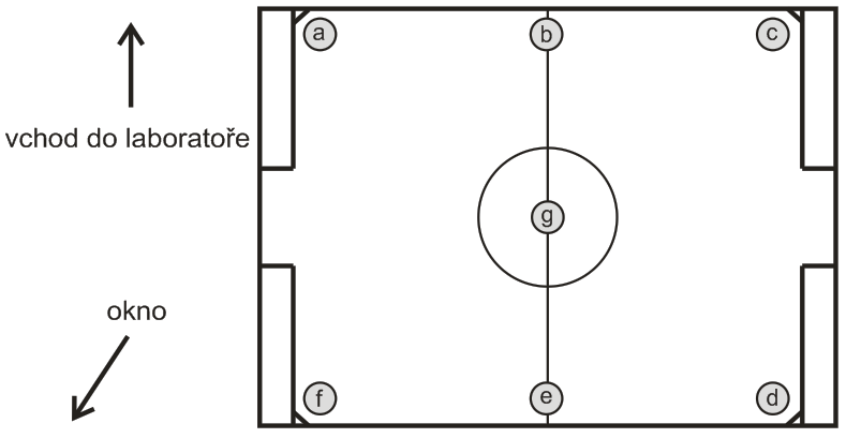
Obrázek 3.3: Umístění kamery nad hřištěm

3.1.1 Osvětlení hřiště

Osvětlení hřiště lze rozdělit na tři základní zdroje:

1. Systém zářivek nad hřištěm – tento systém tvoří celkem šest zářivek ve dvou řadách po třech místěných nad hřištěm.
2. LED diody – jedná se červené o LED diody, které jsou umístěny v blízkosti objektivu.
3. Ostatní zdroje viditelného záření – mezi tyto zdroje je možné zařadit například venkovní světlo.

Na Obrázku 3.4 jsou znázorněny body, ve kterých byla měřena intenzita osvětlení při zapnutém systému zářivek nad hřištěm.



Obrázek 3.4: Znázornění bodů, ve kterých bylo měřeno osvětlení

Naměřené hodnoty intenzity osvětlení v těchto bodech udává Tabulka 3.1. Osvětlení bylo měřeno pomocí luxmetru DIGITAL LUXMETER SOLEX SC-200 (s. č. 940598).

Tabulka 3.1: Měření osvětlení v různých bodech hřiště

pozice	E [lux]
a	372
b	344
c	378
d	381
e	376
f	388
g	400

V Tabulce 3.2 jsou uvedeny hodnoty intenzity osvětlení při plném světle (E_2), při zapnuté jedné řadě zářivek (E_1) a při vypnutých zářivkách (E_0). Intenzita osvětlení byla měřena v bodě „g“ (viz Obrázek 3.4).

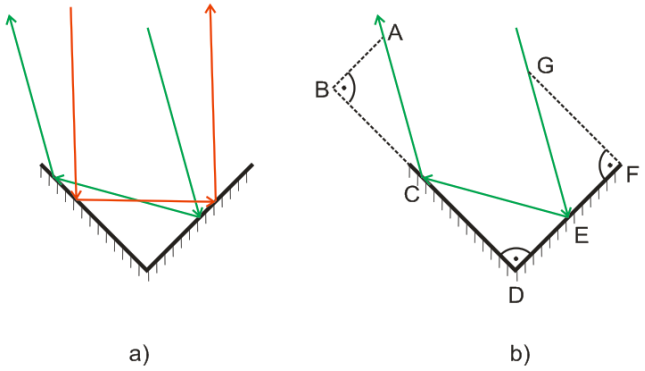
Tabulka 3.2: Měření intenzity osvětlení při různých světelných podmínkách

č. m.	E_0 [lux]	E_1 [lux]	E_2 [lux]
1	15	189	385
2	15	194	400
3	15	195	403
4	15	196	403
5	15	195	401

Nedílnou součástí scény jsou roboty. Pro invariantnost vůči měnícím se světelným podmínkám jsou roboty opatřeny mikroprismatickou reflexní fólií, která obsahuje koutové odražeče.

3.1.1.1 Mikroprismatické fólie

Jedná se o druh fólií, jejichž úkolem je odrážet světlo, které na ně dopadá, zpět ke světelnému zdroji. Fungují na principu tzv. mikrokoutových odražečů, jejichž princip je popsán na Obrázku 2.3. Na tomto obrázku je vidět, že světelné paprsky, které dopadají na mikrokoutový odražeč jsou odráženy zpět, a to rovnoběžně s příchozím paprskem.



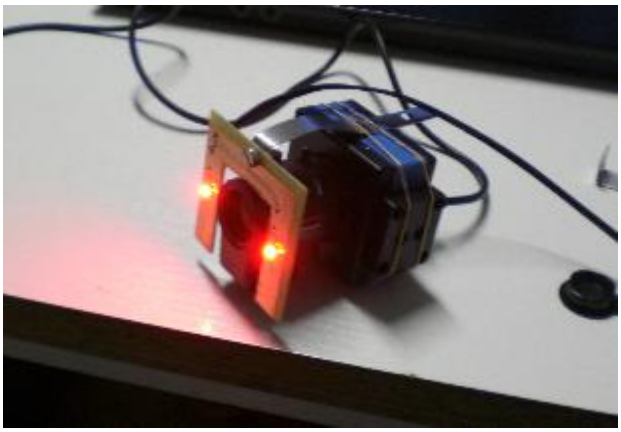
Obrázek 3.5: Princip koutových odražečů

Pokud je koutový odražeč zkoumán v oboru paprskové (geometrické) optiky, je možné dojít k teoretickému závěru, že výsledný odražený paprsek je rovnoběžný s původním paprskem. Pokud se bude jednat o ideální koutový odražeč, tak bude platit vztah (42):

$$\Delta ABC \approx \Delta CDE \approx \Delta EFG \Rightarrow \overleftrightarrow{AC} \parallel \overleftrightarrow{EG} \tag{42}$$

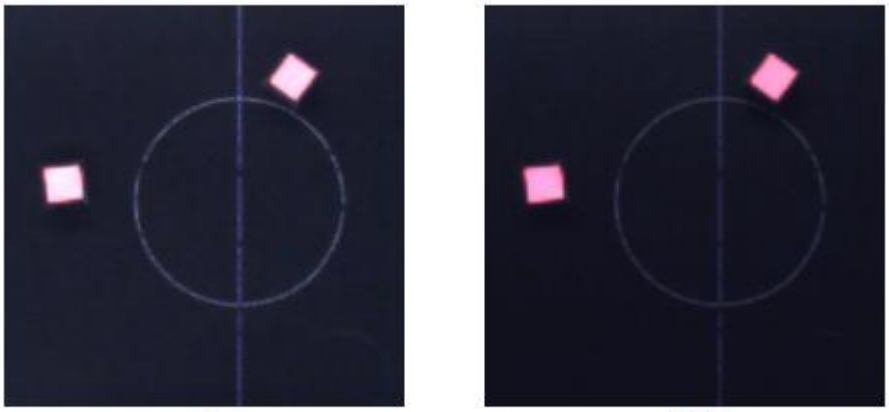
Praktické využití

V diplomové práci byly tyto fólie využity, protože za jakýchkoliv okolních světelných podmínek odráží do kamerového systému stále stejné světlo, které je dané povahou světelného zdroje. Světelný zdroj musí být umístěný co nejblíže optické ose objektivu kamery, aby bylo možné pozorovat paprsky odražené od koutových odražečů fólie. Jako světelný zdroj byly použity dvě SMD LED (typ LSE67B-T2V1-1-1Z). Jedná se o výkonové červené LED diody s vyzařovacím úhlem 120°. Umístění na kameře je zobrazeno na Obrázku 3.6.



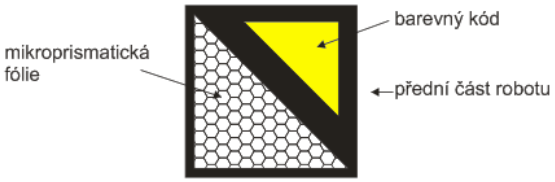
Obrázek 3.6: Umístění LED diod na kameře

Když se horní část robotu opatří reflexní fólií s mikrokoutovými odražeči, tak je možné robot lokalizovat i za zhoršených světelných podmínek. Na Obrázku 3.7a) je znázorněn obraz robotů při rozsvícených dvou řadách zářivek. Na Obrázku 3.7b) je obraz při rozsvícené jedné řadě zářivek.



Obrázek 3.7: Porovnání obrazů z kamery: a) Při rozsvícených obou řadách zářivek, b) Při rozsvícené jedné řadě zářivek

Výsledné označení robotů je uvedeno na Obrázku 3.8. Značení robotu tvoří trojúhelník z mikroprismatické fólie a menší trojúhelník s barevným kódem.



Obrázek 3.8: Označení robotu

3.2 Kamera DFK 21BUC03

Jako snímací prvek byla použita kamera od firmy *The Imaging Source*. Jedná se o barevnou kameru s *CMOS* čipem od firmy *Micron*. K počítači je připojena pomocí sběrnice *USB*. V Tabulce 3.3 jsou uvedeny základní parametry kamery. Fotografie kamery je zobrazena na Obrázku 3.9.

Obrázek 3.9: Kamera DFK 21BUC03 s objektivem Honeywell

Tabulka 3.3: Vlastnosti kamery

Vlastnost	Hodnota
Typ	DFK 21BUC03
Druh snímacího čipu	Micron CMOS 1/3 “
Rozlišení čipu	744 × 480 pixelů
Rychlost snímání	5; 7,5; 15; 30; 60 fps
Barevné režimy	RGB, BY8

Spolu s kamerou je dodáván software *IC Capture*, který slouží k jednoduchému nastavování parametrů kamery, snímání obrazu a videa. Dále je dodáván balíček *IC Imaging Control*, který obsahuje knihovny a komponenty pro práci s kamerou. Při testování kamery bylo jištěno, že při rychlosti snímání 30 fps je snímek silně zarušený (viz Obrázek 3.10). Toto zarušení nastávalo pouze při rychlosti snímání 30 fps, při ostatních rychlostech snímání se tato závada neobjevila. Původ závady nebyl zjištěn. Kamera byla testována na různých počítačích i s různými propojovacími kabely, ale za všech podmínek se vada projevovala stejně. Při snímání kamerou bylo dále zjištěno, že je citlivá na kolísání napájecího napětí. I když je napájena přes *USB*, tak například při zapnutí zářivky kamera „zmrzla“. Po sepnutí vypínače nebylo možné s kamerou komunikovat. Příčina je pravděpodobně v tom, že novodobé zářivky vracejí zpětný ráz do sítě. Tento ráz projde i skrz napájecí zdroj notebooku a přes *USB* až do kamery. Při vypojení notebooku z elektrické sítě výpadek nenastával. Součástí snímacího zařízení je kromě kamery i objektiv od firmy *Honeywell*, který disponuje manuálním ostřením, clonou a zvětšením. Jedná se o typ *HLM28V8F95*.

Obrázek 3.10: Porovnání výstupů z kamery: a) Zašuměný obraz při 30 fps, b) Obraz při 60 fps

3.2.1 Vliv šumu na výstupní obraz

Z důvodu zjištění, jaký má vliv šum na výstupní obraz kamery, bylo provedeno měření, které se skládalo z jednoduchého postupu. Nejdříve byl zakryt objektiv pomocí krytky tak, aby na čip pronikalo co nejméně světla. Potom byla snímána série snímků, které byly poté vyhodnoceny. V ideálním případě by měly být hodnoty všech složek u všech pixelů nulové, ale takový

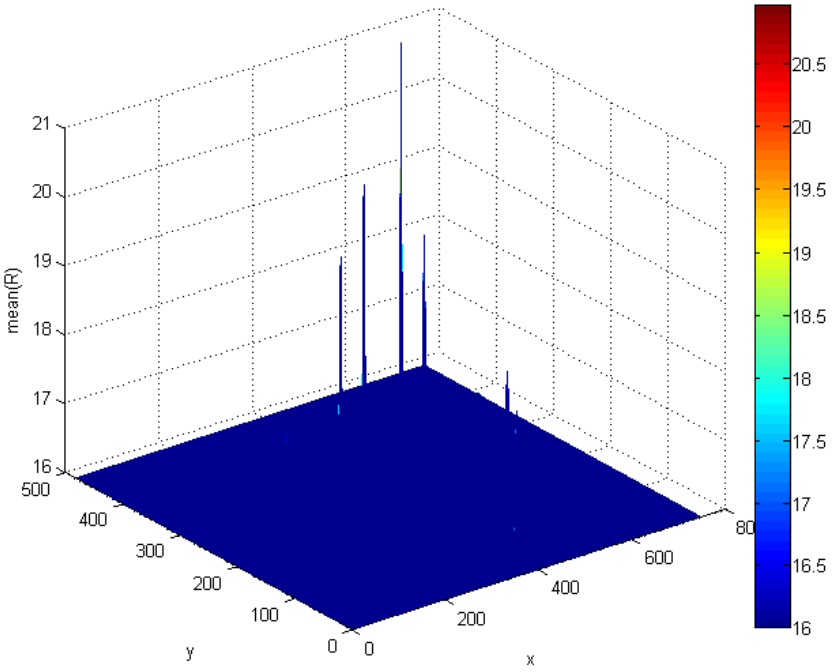
případ nikdy nenastane. Po dobu sta sekundy byl každou sekundu snímán obraz. Těchto sto snímků bylo poté vyhodnoceno. Kamera měla vypnuté automatické vyvážení bílé a nastavenou expozici na 1/64 s. Ke každému pixelu se počítala průměrná hodnota jasu u konkrétní složky. Průměr byl počítán podle vztahu (43):

$$\overline{L_R}(x,y) = \frac{1}{n}\sum_{i=1}^{n} f_{R,i}(x,y), \tag{43}$$

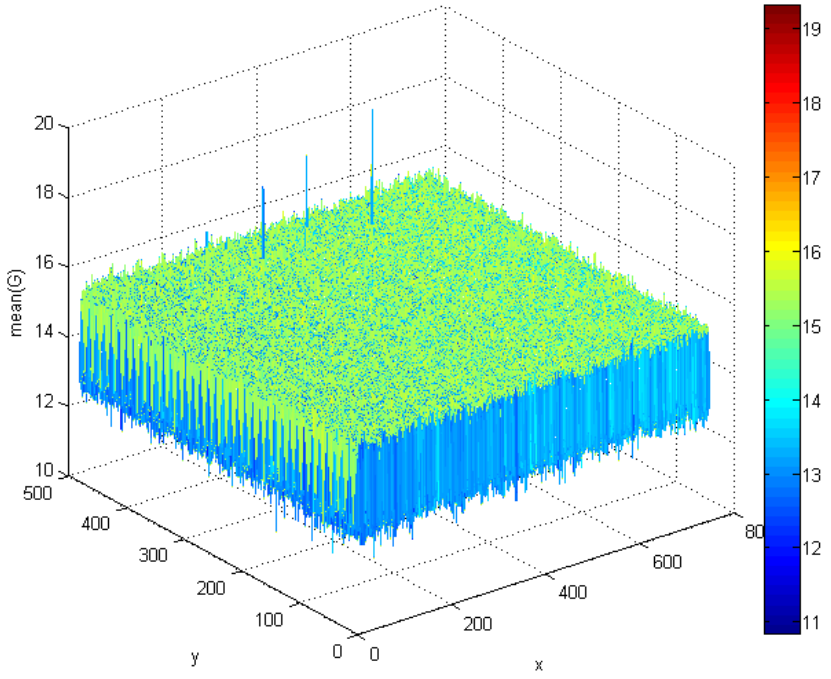
kde $\overline{L_R}(x,y)$ průměrná hodnota jasu červené složky pixelu o souřadnicích x a y, n je počet nasnímaných obrazů a $f_{R,i}(x,y)$ je hodnota jasu červené složky i-tého obrazu o souřadnicích pixelu x a y. Výsledky průměrných hodnot jasu jsou pro konkrétní složky zobrazeny pomocí grafu na Obrázku 3.11 - 3.13. Výběrová variance byla počítána pro každý pixel a složku podle vzorce (44):

$$s_R^2(x,y) = \frac{1}{n-1}\sum_{i=1}^{n}\left[f_{R,i}(x,y) - \overline{L_R}(x,y)\right]^2, \tag{44}$$

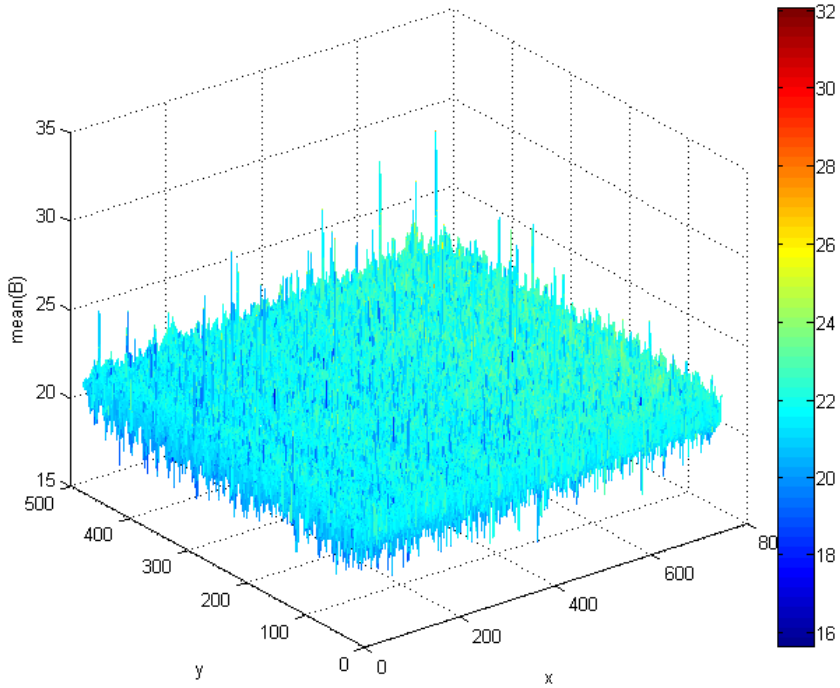
kde $s_R^2(x,y)$ je výběrové variance jasu červené složky pixelu o souřadnicích x a y. Grafické vyjádření rozptylu pro konkrétní body je zobrazeno pro konkrétní složky na Obrázku 3.14 – 3.16.



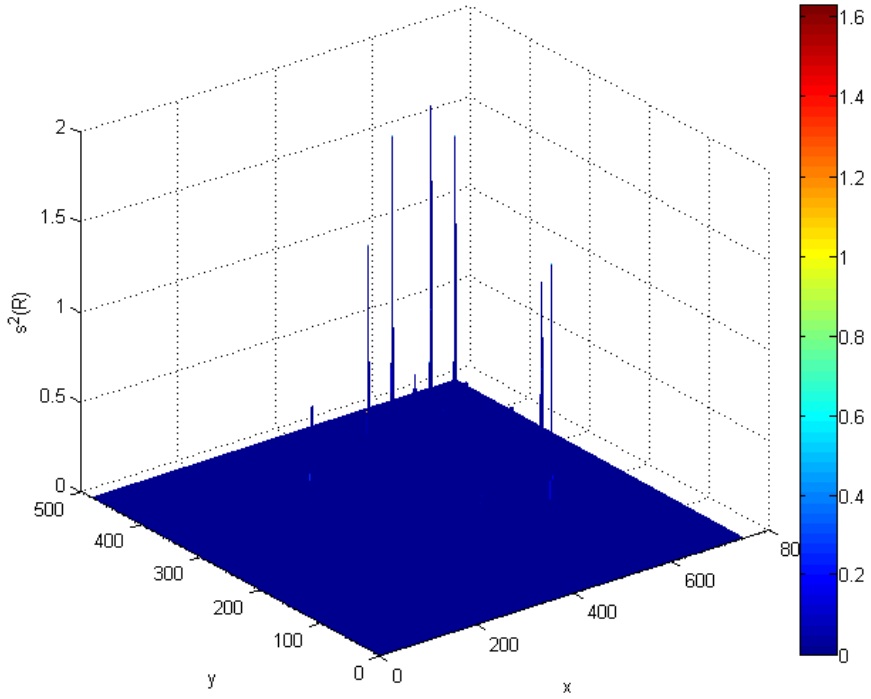
Obrázek 3.11: Průměrná hodnota jasu u červené složky obrazu v závislosti na poloze bodu



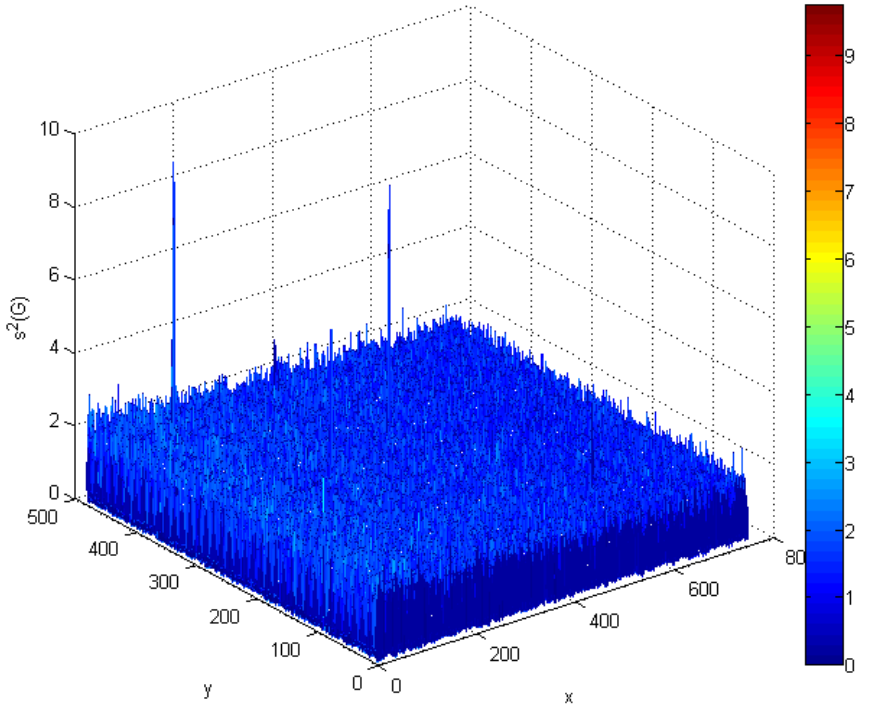
Obrázek 3.12: Průměrná hodnota jasu u zelené složky obrazu v závislosti na poloze bodu



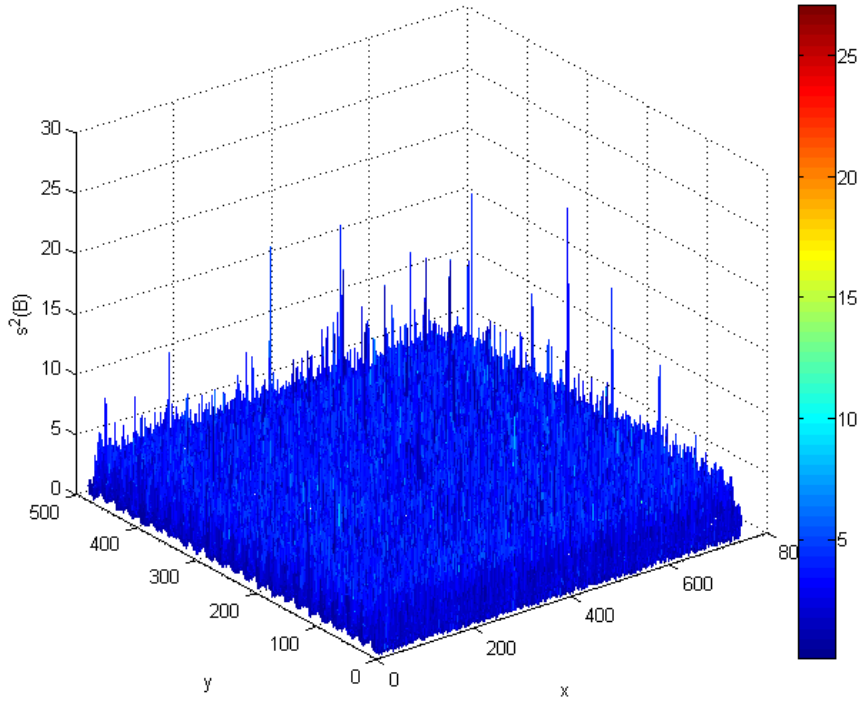
Obrázek 3.13: Průměrná hodnota jasu u modré složky obrazu v závislosti na poloze bodu



Obrázek 3.14: Výběrová variance hodnot jasu u červené složky obrazu v závislosti na poloze bodu



Obrázek 3.15: Výběrová variance hodnot jasu u zelené složky obrazu v závislosti na poloze bodu



Obrázek 3.16: Výběrová variance hodnot jasu u modré složky obrazu v závislosti na poloze bodu

Z výše uvedených grafů vyplývá, že červená složka je nejméně zatížená šumem (viz Obrázek 3.11 a 3.14). Průměrná hodnota jasu u červené složky je téměř konstantní a rozptyl se u většiny bodů blíží k nule. Proto lze prohlásit, že výstupní obraz červené složky je zatížen pouze systematickou chybou, kterou by bylo možné v případě potřeby odstranit.

Modrá složka byla nejvíce zarušená. Průměrné hodnoty jasu modré složky v závislosti na poloze jsou velmi různorodé. I hodnoty rozptylu (viz Obrázek 3.16) byly nesrovnatelně vyšší než u zelené či červené složky.
V důsledku těchto faktů bylo rozhodnuto, že pro identifikaci robotů se budou používat mikroprismatické fólie, na které bude svítit červené světlo a analyzovat se bude pouze červená složka obrazu.

3.2.2 Radiální zkreslení

Při práci s kamerou bylo zjištěno, že v důsledku nedokonalosti objektivu, se objevuje radiální zkreslení. Na Obrázku 3.17 je obraz z kamery, který je zatížený výše zmíněným zkreslením. Z tohoto snímku je patrné, že se jedná o tzv. soudkovitost, protože zvětšení ve středu obrazu je vyšší než na jeho okrajích. Originální nezkreslený obraz je pravoúhlá mřížka.

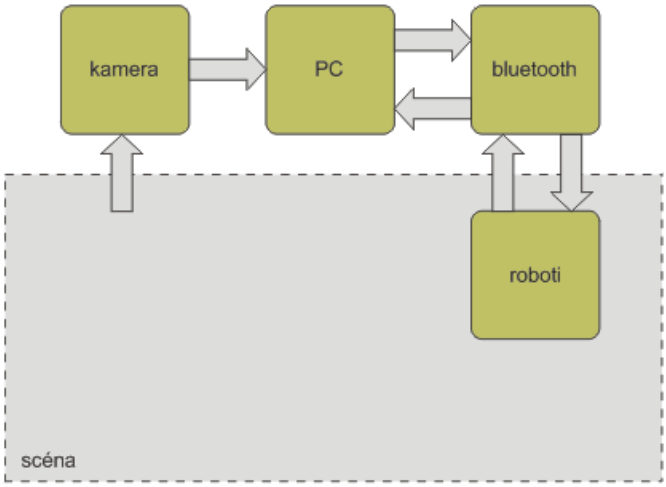
Obrázek 3.17: Ukázka zkresleného obrazu

Na odstranění zkreslení byly použity funkce knihovny *OpenCV*, pomocí kterých byly identifikovány jednotlivé parametry. Pomocí vztahu (45) je popsáno radiální zkreslení.

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = (1 - 1{,}032r^2 + 4{,}377r^4 - 16{,}300r^6) \cdot \begin{bmatrix} x \\ y \end{bmatrix}, \tag{45}$$

3.3 Návrh softwarového řešení

Celý návrh softwarového řešení musí brát ohled na hardwarové sestavení systému. Systém se skládá z pěti základních částí, které jsou zobrazeny na Obrázku 3.18. Mezi tyto části patří kamera, počítač, bluetooth, roboti a scéna. Úkolem monitorovacího systému je zjišťovat aktuální polohu robotů a umožňovat jejich identifikaci. V tomto případě bude monitorovací systém zajišťovat získání informace o absolutní pozici konkrétního robotu.



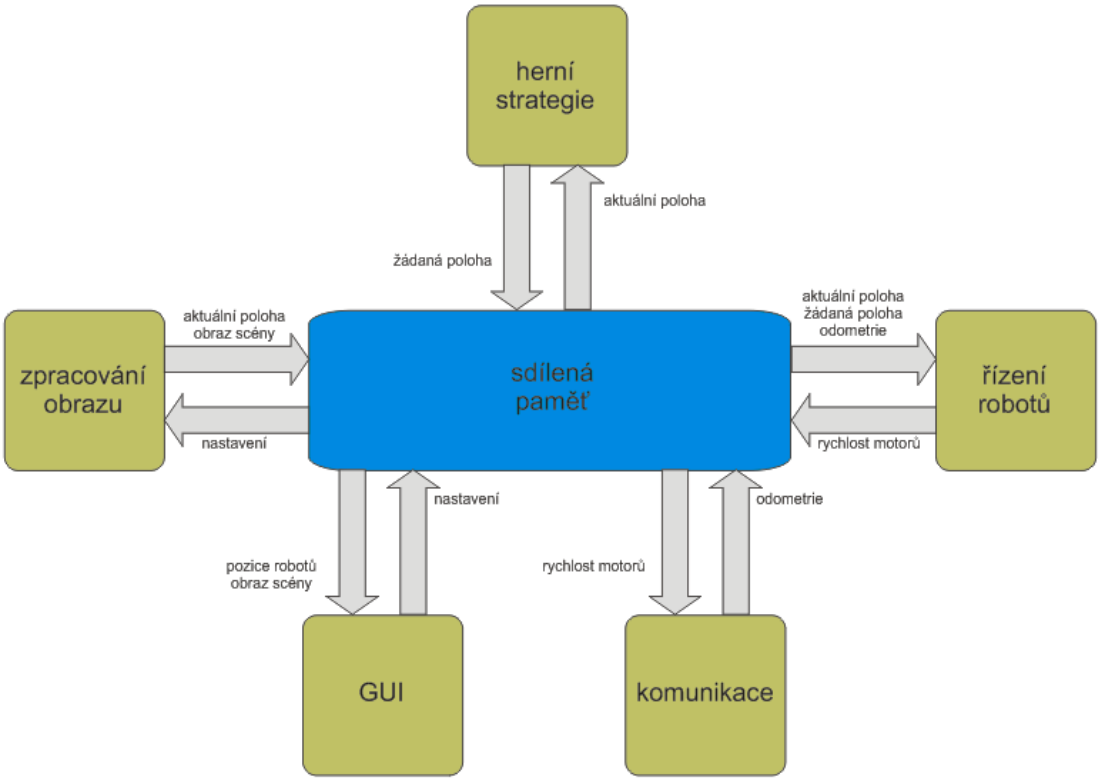
Obrázek 3.18: Hardwarové schéma projektu

Jedním ze základních požadavků na software, který bude identifikovat a určovat polohu robotů, je rychlost a modulárnost systému. Pod pojmem

rychlost, se skrývá mnohem přísnější kritérium, jímž je real-timová odezva. Tedy aby systém byl schopen vyhodnotit obraz během určeného časového intervalu, který bude výrazně menší, než bude perioda vzorkování (1/30 nebo 1/60 s), protože počítač musí během této doby obsloužit i jiné moduly, než je zpracování obrazu.

3.3.1 Modulárnost systému

Jak již bylo napsáno výše, monitorování mobilních jednotek bude tvořit jeden z mnoha modulů celého softwarového řešení. Předpokládaná struktura jednotlivých modulů je zobrazena na Obrázku 3.19. Hlavním úkolem modulu monitorování mobilních jednotek (dále jen modul „Zpracování obrazu“) je zpracovat obraz a z něj identifikovat jednotlivé objekty (roboty a míč) a jejich pozice.



Obrázek 3.19: Softwarové schéma

3.3.2 Prostředky meziprocesové komunikace

Jak je patrné z Obrázku 3.18, jednotlivé moduly mezi sebou komunikují a sdílejí data pomocí tzv. sdílené paměti, která patří mezi prostředky meziprocesové komunikace. Při práci se sdílenou pamětí je nutné zajistit, aby do paměti přistupoval pouze jeden proces. Unikátní přístup je možné zajistit pomocí synchronizačních objektů, mezi něž patří například mutexy, semafory, eventy atd. Při výběru vhodné metody pro realizaci sdílené paměti je

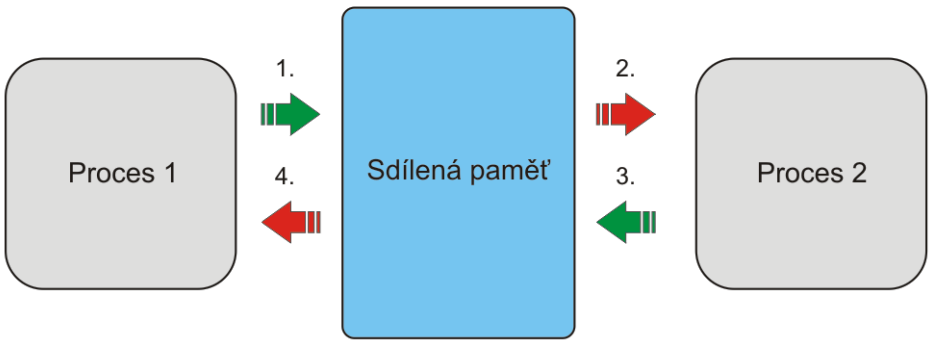
rozhodujícím faktorem datová propustnost. V případě zpracování obrazu se bude jednat o datový tok přibližně 61 MBps (3*744*480*60) při 60 fps a 30 MBps při 30 fps, pokud jsou zanedbány údaje o poloze robotů, které se pohybují v řádech desítek kBps.

3.3.2.1 CreateFileMapping – testování propustnosti

Test probíhal pomocí dvou procesů a sdílené paměti vytvořené pomocí *CreateFileMapping*. Nejdříve se pomocí *CreateFileMapping* vytvoří sdílená paměť a poté se s využitím funkce *MapViewOfFile* získá ukazatel na sdílenou paměť. Mezi procesy se předává datový blok v paměti o velikosti 1 MB, tento blok se přesouvá do sdílené paměti pomocí funkce *CopyMemory*. Test probíhal podle Obrázku 3.20 ve čtyřech základních fázích:

1. Proces1 zkopíruje úsek v paměti o velikosti 1 MB na sdílenou paměť. Inkrementuje Semafor1.
2. Proces2 čeká, až bude Semafor1 v signálním stavu. Poté zkopíruje data ze sdílené paměti do statické proměnné.
3. Proces2 zkopíruje obsah mírně pozměněné lokální statické proměnné do sdílené paměti a inkrementuje Semafor2.
4. Proces1 čeká, až bude Semafor1 v signálním stavu. Poté zkopíruje data ze sdílené paměti do lokální proměnné.

Bod 1. – 4. se cyklicky 1000× opakuje. Z naměřené doby trvání tohoto cyklu se určí přibližná datová propustnost.



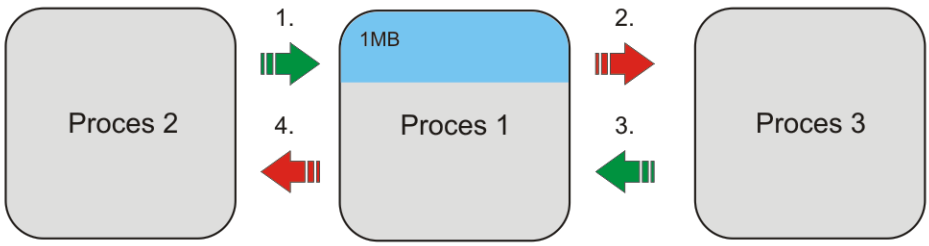
Obrázek 3.20: CreateFileMapping – schéma testování propustnosti

Zhodnocení a porovnání s metodou *ReadProcessMemory* je v kapitole 3.3.2.3.

3.3.2.2 ReadProcessMemory – testování propustnosti

Funkce *ReadProcessMemory* a *WriteProcessMemory* umožňují číst popřípadě zapisovat do paměti jiného procesu, pokud známe adresu, na které daný blok dat leží. Test probíhal pomocí tří procesů, z nichž první poskytoval svoji paměť pro sdílení dat. Ostatní dva procesy si mezi sebou vyměňovaly data

pomocí paměti prvního procesu. Test probíhal podle Obrázku 3.21 ve stejných fázích jako u *CreateFileMapping*.



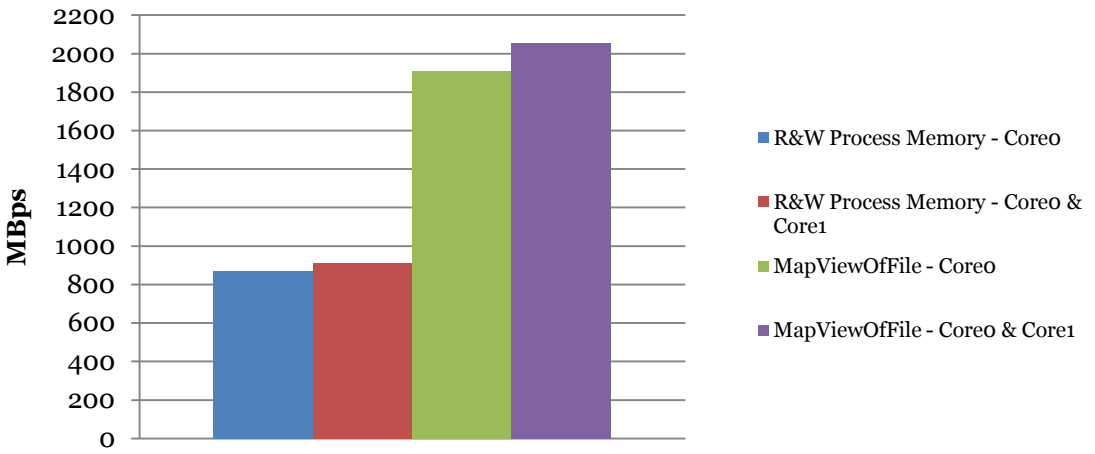
Obrázek 3.21: ReadProcessMemory - schéma testování propustnosti

3.3.2.3 Porovnání metod

Naměřené průměrné hodnoty datové propustnosti jsou uvedeny v Tabulce 3.4. Grafické znázornění těchto hodnot je na Obrázku 3.22.

Tabulka 3.4: Porovnání propustnosti pro různé metody

Metoda sdílení	Propustnost [MBps]
R&W Process Memory - Core0	868,6
R&W Process Memory - Core0 & Core1	909,5
MapViewOfFile - Core0	1911,2
MapViewOfFile - Core0 & Core1	2056,3



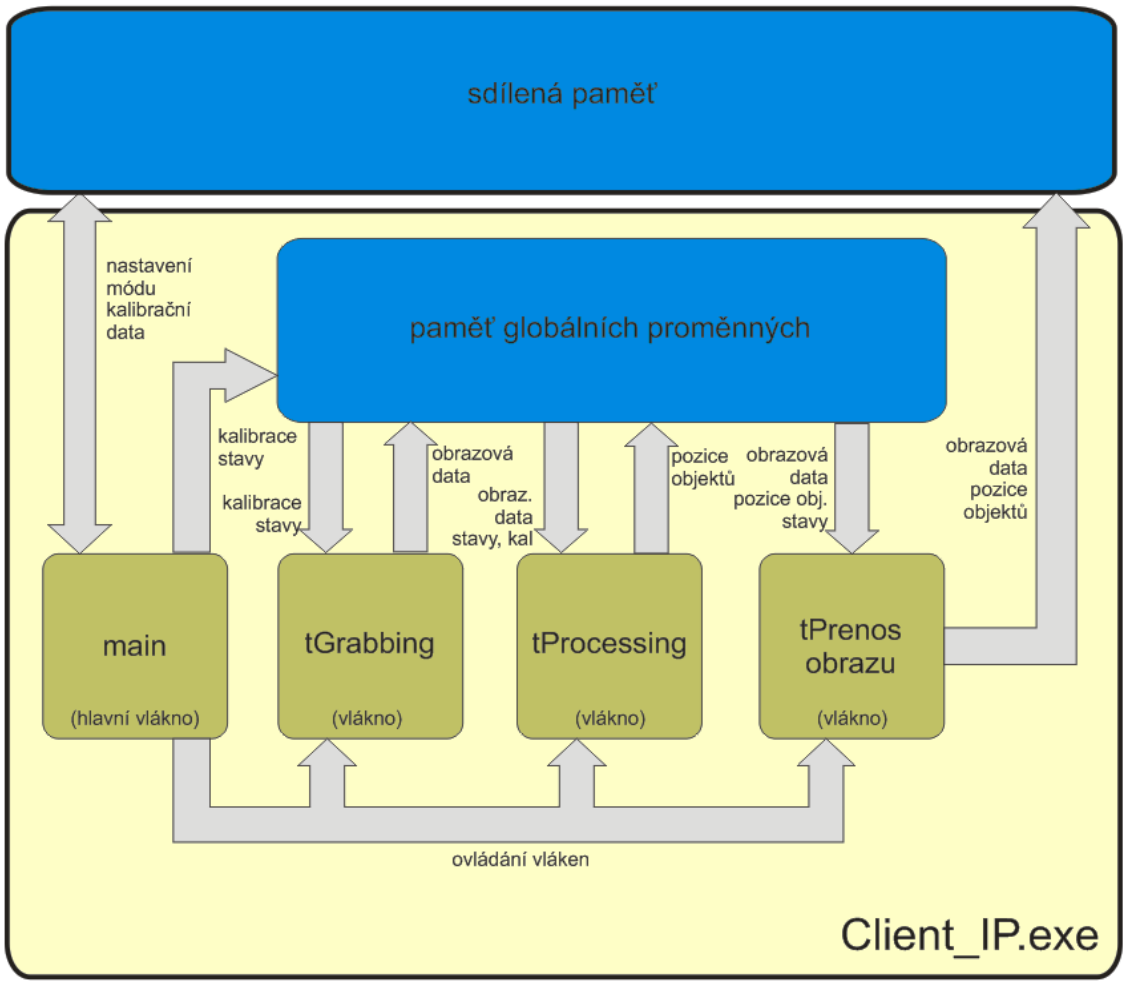
Obrázek 3.22: Grafické porovnání datových propustností

Z grafu na Obrázku 3.22 je vidět, že datová propustnost při využití metody *CreateFileMapping* je přibližně dvakrát vyšší, než je tomu u metody *WriteProcessMemory*. *CreateFileMapping* má výhodu v tom, že jediné, co potřebují jednotlivé procesy znát, je název sdílené paměti. Tento název se nikdy nemění (v rámci projektu má jedinečnou hodnotu). Naproti tomu u *WriteProcessMemory* je nutné znát adresy daných proměnných v paměti, které jsou při každém spuštění odlišné. Z důvodu vyšší propustnosti a lepšímu přístupu byla vybrána metoda *CreateFileMapping*.

3.3.3 Počítačové zpracování obrazu

Zpracování obrazu bude probíhat s využitím knihovny *OpenCV*. Jedná se o multiplatformní otevřenou knihovnu zaměřenou na počítačové vidění. Je napsána v jazyce *C* a *C++* a je možné ji využívat na různých operačních systémech (např. *Windows*, *Linux* a *Mac OS X*). V současnosti je aktivní vývoj podporující využívání *OpenCV* v různých programovacích prostředích (*Python*, *Ruby*, *Matlab* a jiné jazyky). Knihovna byla navržena s důrazem na co největší výpočetní efektivitu a je silně zaměřena na real-time aplikace(Bradski, a další, 2008).

Aplikace, která má na starosti zpracování obrazu, je navržena ve vývojovém prostředí Microsoft *Visual C++ 2008*. Strukturu této aplikace je možné ve zjednodušené formě popsat pomocí Obrázku 3.23.



Obrázek 3.23: Schéma aplikace Client_IP.exe

Aplikace pro zpracování obrazu (*„Client_IP.exe“*) obsahuje čtyři základní vlákna:

1. **main** – jedná se o hlavní vlákno, které řídí celou aplikaci. Toto vlákno periodicky načítá „mód“ ze sdílené paměti a podle něj ovládá jednotlivá

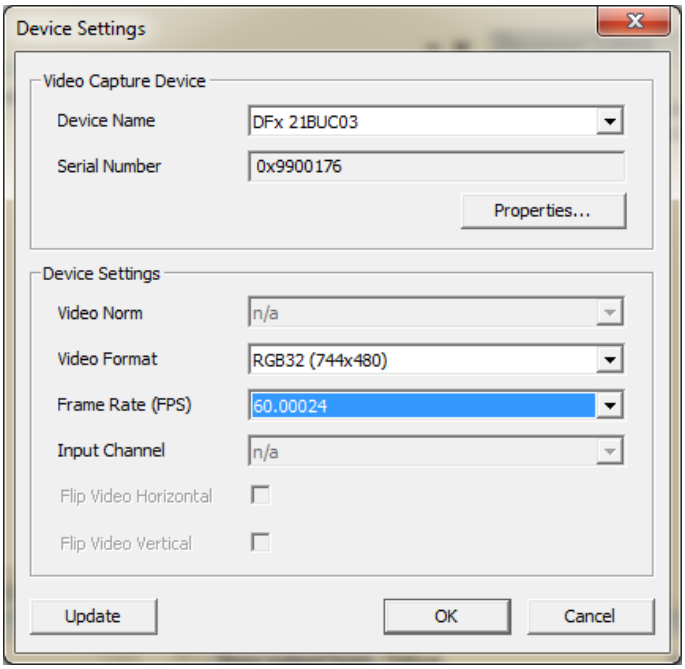
vlákna. Z tohoto vlákna jsou také volány funkce umožňující kalibraci a ostření kamery.

2. **tGrabbing** – toto vlákno má nejvyšší prioritu a obstarává získání snímku z kamery, jeho ozrcadlení a odstranění zkreslení. Vlákno ukládá do globálních struktur obrazová data spolu s časovým razítkem.
3. **tProcessing** – hlavním úkolem vlákna je zpracovávat obraz, tedy určit polohy a natočení jednotlivých robotů a rozpoznat je od ostatních objektů. Data o objektech ukládá do globálních struktur. Vlákno má střední prioritu. Toto vlákno vykonává předzpracování, segmentaci, popis, klasifikaci, určení natočení, určení pozice a identifikaci robotů. Celý tento jeden cyklus trvá v průměru 11 ms na notebooku, jehož konfigurace je uvedena v příloze.
4. **tPrenosObrazu** – je vlákno, které zajišťuje přenos obrazu a údajů o objektech do sdílené paměti. Vlákno má nejnižší prioritu.

Všechna vlákna přistupují ke globálním proměnným pomocí synchronizačních prvků, aby nedocházelo k možným kolizím.

3.3.3.1 Přístup ke kameře

V knihovně *OpenCV* jsou funkce, podporující práci s kamerou. Tyto funkce však nemohly být využity, protože pomocí nich nebylo možné nastavit rychlost snímání a barevný formát. Implicitní rychlost snímání kamery je nastavena na 30 fps. Při této rychlosti snímání je obraz silně zarušen, proto se ke kameře přistupovalo pomocí knihoven, které dodává výrobce. Pomocí těchto knihoven je možné zobrazit dialog pro nastavení kamery. Dialog pro nastavení kamery je zobrazen na Obrázku 3.24.

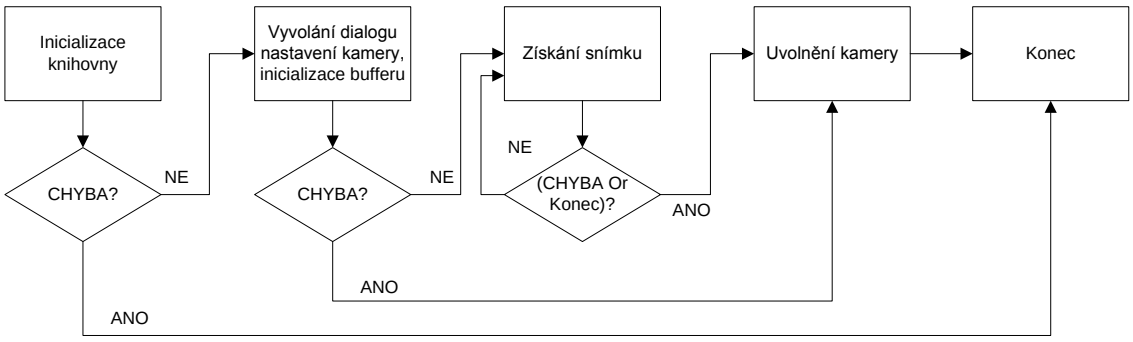


Obrázek 3.24: Dialog pro nastavení kamery

Pro další zpracování bylo nutné kopírovat data z bufferu výrobce do *OpenCV* struktury *IplImage*. Datová struktura obou struktur je velmi podobná, rozdíl je v pouze v chápání řazení bytů, proto je obraz zkopírovaný z bufferu do struktury *IplImage* vertikálně převrácený. Funkce, které přistupují ke kameře, jsou definovány v modulu *„grabbing.cpp“*. Mezi nejdůležitější funkce patří:

- *BOOL mInicializaceKnihovny(char * serioveCislo)* – inicializuje knihovny dodávané výrobcem kamery
- *IplImage * mInicializaceKamery(void)* – vyvolá dialog pro nastavení kamery a inicializuje datové struktury podle nastavení kamery
- *BOOL mUvolneniKamery(void)* – ukončí a uvolní kameru
- *BOOL mSnapImage(IplImage * image)* – získá snímek z kamery
- *DWORD WINAPI tGbrabbing(LPVOID lpParam)* – vlákno, které periodicky ukládá obrazová data do globálního bufferu

Snímání obrazu obstarává samostatné vlákno, které periodicky kopíruje do bufferu obrazová data určená k dalšímu zpracování. Ve zjednodušené formě by se proces snímání dal popsat pomocí vývojového diagramu na Obrázku 3.25.



Obrázek 3.25: Proces snímání

3.3.3.2 Kalibrace kamery

Kalibrace kamery probíhala pomocí vnitřních funkcí *OpenCV*, které umožňují eliminovat radiální zkreslení. Alogritmus použitý v diplomové práce je podrobně popsán v [3]. Ke kalibraci si používá černobílá šachovnice na tvrdém podkladu. Algoritmus lze popsat pomocí následujících kroků:

3. Nalezení počtu rohů šachovnice v obraze pomocí funkce *cvFindChessboardCorners(…)*
4. Zpřesnění pozice jednotlivých rohů se subpixelovou přesností pomocí funkce *cvFindCornerSubPix(…)*
5. Vykreslení nalezených rohů funkcí *cvDrawChessboardCorners(…)*
6. Výpočet kalibračních matic A a B pomocí funkce *cvCalibrateCamera2(…)*, kde A je matice vlastností kamery a B je matice parametrů zkreslení – viz vztah (46).

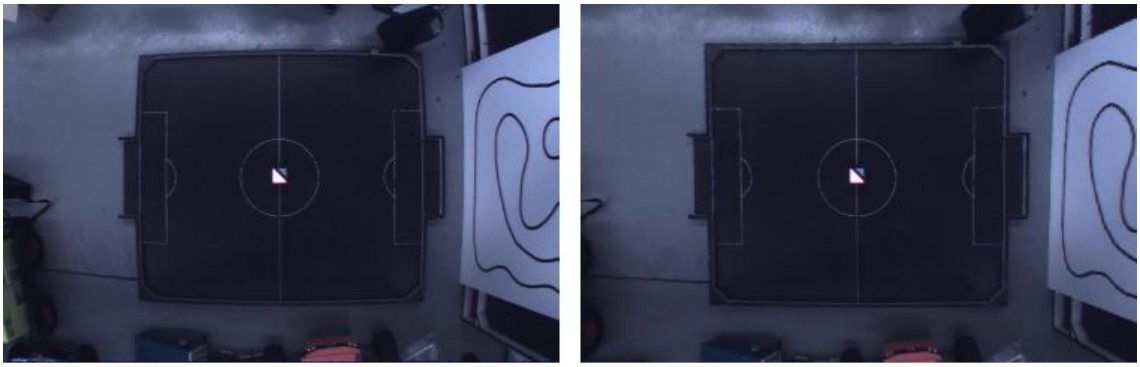
$$A = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}, \qquad B = [k_1 \quad k_2 \quad p_1 \quad p_2 \quad k_3] \tag{46}$$

1. Inicializace struktury, která obsahuje přetransformované body, pomocí *cvInitUndistortMap(...)*.

Na Obrázku 3.26 je snímek z kamery, na kterém předchozí algoritmus vyznačil nalezené rohy šachovnice.

Obrázek 3.26: Ukázka nalezení rohů šachovnice

Ukázka originálního snímku a snímku s transformovanými souřadnicemi a interpolovanými jasovými hodnotami je na Obrázku 3.27. Jako interpolační metoda byla použita metoda nejbližšího souseda.



a) b)

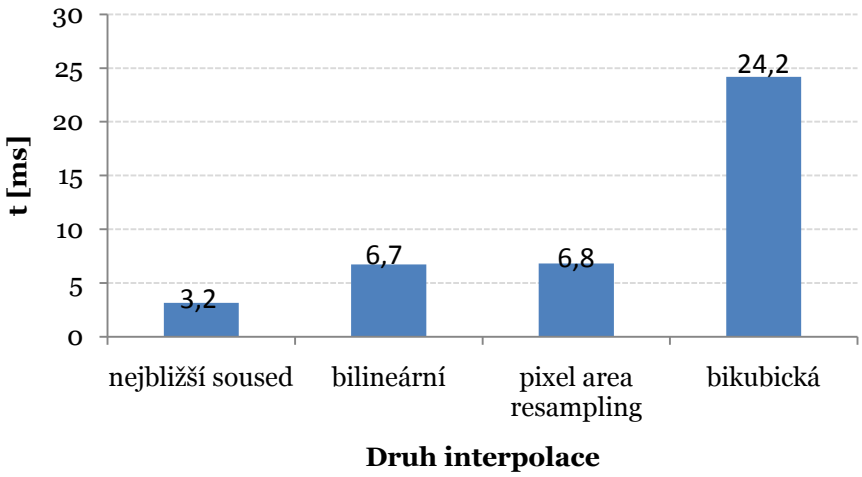
Obrázek 3.27: Porovnání obrazů: a) Originální obraz, b) Obraz s transformovanými souřadnicemi a interpolovanými hodnotami jasu

Testování výpočetní náročnosti interpolačních metod

Protože k interpolaci barevných hodnot dochází každou periodu, je nutné získat přehled o tom, jak jsou jednotlivé metody výpočetně náročné. Použitá knihovna *OpenCV* obsahuje funkci *cvRemap*, která umožňuje interpolaci pomocí nejbližšího souseda, bilineární, bikubickou a tzv. pixel area resampling. Test probíhal tak, že se měřila doba vykonání pět-seti interpolací pro každou metodu. Poté byla tato doba přepočítaná na dobu trvání jedné operace. Počítač, na kterém byly metody testovány, má následující hardwarovou konfiguraci

uvedenou v příloze č. 1. Interpolace hodnot barev byla prováděna na diskrétním obrazu s rozlišením 744 × 480 pixelů a barevným modelem RGB.

Na Obrázku 3.28 je zobrazen graf, ve kterém jsou porovnány jednotlivé interpolační metody. Z grafu je patrné, že nejrychlejší metodou je nejbližší soused.



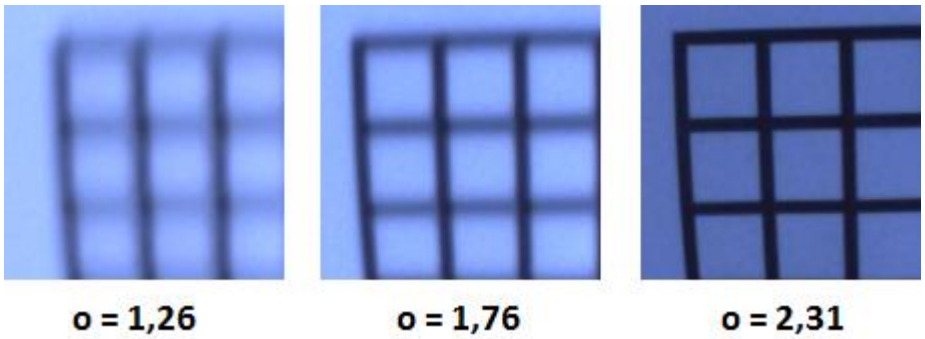
Obrázek 3.28: Porovnání výpočetní náročnosti jednotlivých interpolačních metod

3.3.3.3 Ostření kamery

Pro účely co nejlepšího zaostření byla naprogramována funkce *mOstreni(...)*, která se snaží kvantifikovat ostrost obrazu. Výstup funkce je dán vztahem (47):

$$o = 10 \cdot \sum_{i=0}^{n-1} \sum_{j=0}^{m-1} \frac{|g_x(i,j)| + |g_y(i,j)|}{f_g(i,j)}, \tag{47}$$

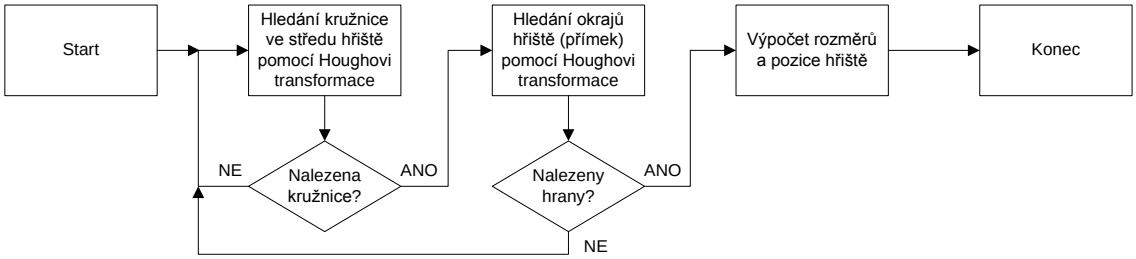
kde o je kvantitativní míra ostrosti obrazu, $g_x(i,j)$ diference v ose x, $g_y(i,j)$ diference v ose y a $f_g(i,j)$ vstupní šedotónový obraz. Příklad výstupních hodnot funkce je uveden na Obrázku 3.29, na kterém je možné pozorovat vzrůstající hodnotu o s ostrostí obrazu.



Obrázek 3.29: Příklad výstupu funkce *mOstreni* pro různé vstupní obrazy

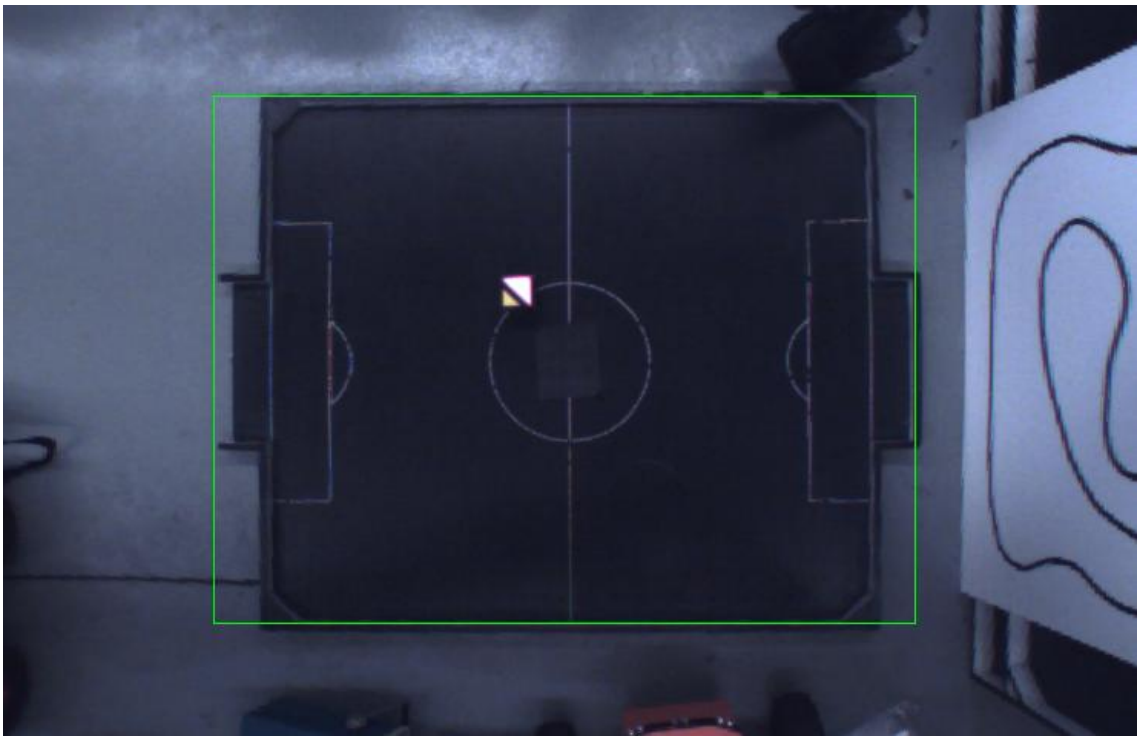
3.3.3.4 Nalezení hřiště

Aby mohla být korektně určena poloha robotů, je nutné nejdříve nalézt, kde se v obraze nachází hřiště. Aby bylo možno hřiště nalézt, je nutné zkoumat obraz, u kterého je odstraněno radiální zkreslení. Zjednodušený princip algoritmu pro hledání hřiště je zobrazen na Obrázku 3.30.



Obrázek 3.30: Princip algoritmu pro hledání hřiště

Nejdříve je pomocí Houghovy transformace nalezena kružnice, která je uvnitř hřiště. Kružnice je hledána v celém obraze pro $r \in \langle 30; 60 \rangle$ pixelů. Po nalezení středu kružnice jsou známy oblasti, v kterých se mohou nacházet jednotlivé hrany hřiště. Pokud jsou nalezeny všechny čtyři hrany hřiště, tak se pomocí jejich rovnic vypočítá oblast, ve které se pravděpodobně nalézá hřiště. Ukázka nalezení hřiště je zobrazena na Obrázku 3.31. Výše zmíněný algoritmus je obsažen ve funkci *mNajdiHriste(…)*.



Obrázek 3.31: Ukázka nalezení hřiště

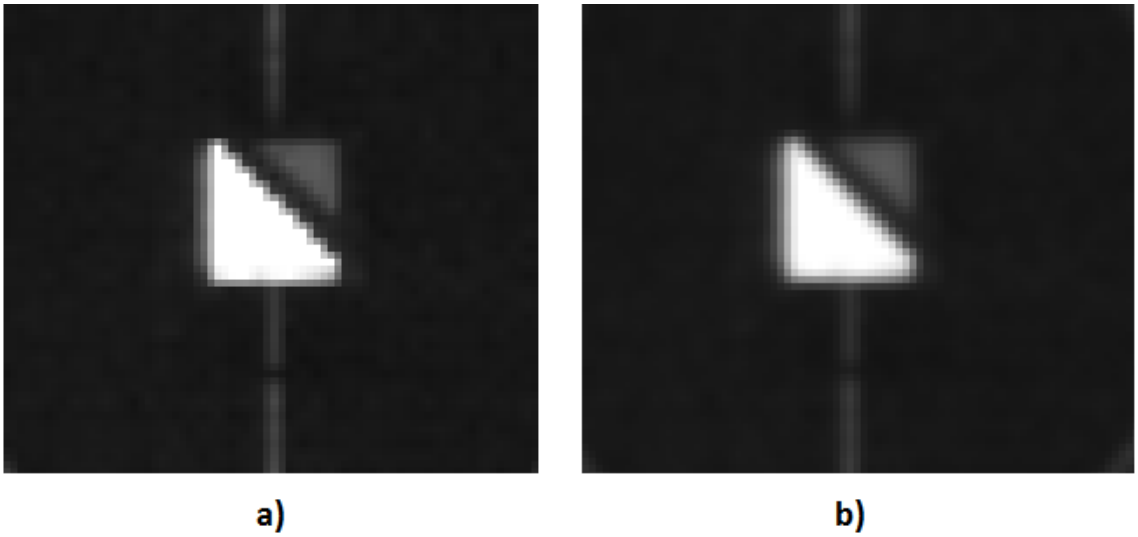
3.3.3.5 Předzpracování obrazu

Předchozí metody (kap. 3.3.3.2 – 3.3.3.4) jsou používány pouze jednorázově. Tedy neopakují se cyklicky během každé periody vzorkování obrazu. Z tohoto důvodu u předchozích metod nebyla řešena výpočetní náročnost. U předzpracování obrazu pro následnou segmentaci a popis je nutné brát v úvahu výpočetní náročnost.

Aby náročnost zpracování byla co nejmenší, tak se zpracovává pouze region, který je vytyčen hřištěm. V předzpracování se neodstraňuje geometrické zkreslení ze dvou důvodů:

1) Interpolace jasových úrovní transformovaných bodů způsobuje nespojitosti jasové funkce a tím dochází v mnoha případech k přerušení hran. Tento fakt je pro segmentaci z hranové reprezentace nežádoucí.
2) Výpočetní náročnost interpolačních algoritmů je poměrné vysoká. Téměř vždy se jedná o určitý kompromis mezi výpočetní náročností a kvalitou výstupního obrazu.

Při testování bylo jištěno, že na kvalitu segmentace má pozitivní vliv použití filtru s Gussovým rozložením. Tento fakt je o to zajímavější, že filtr s Gaussovým rozložením je volán i při segmentaci uvnitř Cannyho hranového filtru. Aplikace filtru s Gaussovým rozložením s konvoluční maskou 3×3 je jedinou metodou předzpracování (kromě převodu na šedótonový obraz), která byla využita. Tento úkon v průměru trval na notebooku (konfigurace viz příloha č. 1) 2 ms. Na Obrázku 3.32 je výřez obrazu před a po aplikaci filtru.

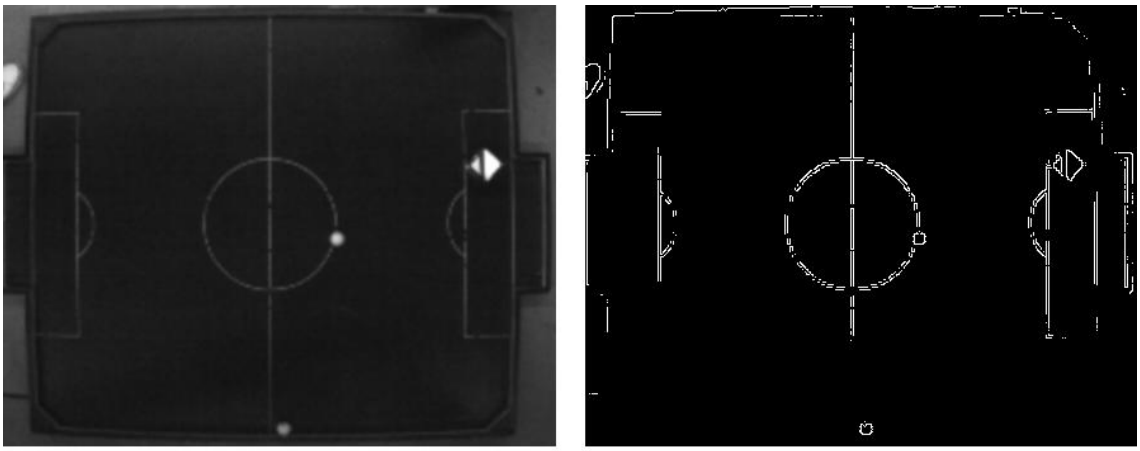


Obrázek 3.32: Ukázka použití filtru s Gaussovým rozložením: a) Originální obraz (červená složka z RGB obrazu), b) Obraz po aplikaci filtru s Gaussovým rozložením

3.3.3.6 Segmentace

Segmentace probíhá na základě hranové reprezentace, které je získána pomocí Cannyho hranové detektoru. Výsledkem segmentace je poté obraz, který

je zobrazen na Obrázku 3.33. Pro nastavení Cannyho hranového detektoru se nejvíce osvědčilo nastavení dolního i horního prahu na 120.



Obrázek 3.33: Ukázka Cannyho hranového detektoru: a) Vstupní šedtónový obraz, b) Hranová reprezentace získaná pomocí Cannyho hranového detektoru

3.3.3.7 Popis objektů

Před hledáním příznaků je nutné jednotlivé objekty do sebe odlišit. Je nutné tedy každému objektu přiřadit index, aby mohl být dále zkoumán. K tomuto úkonu je použita funkce *cvFindContours(…)*. Vstupem do této funkce je binární obraz hranové reprezentace obrazu a výstupem jsou sekvence kontur objektů. Každá sekvence obsahuje ukazatel na první, poslední, předchozí a následující konturu. Dále v sobě obsahuje ukazatel na paměť, kde jsou uloženy souřadnice bodů, které tvoří obrys (konturu) daného objektu. Fakt, že při dalším zpracování je pracováno pouze se souřadnicemi, výrazně urychluje všechny operace. Funkce navíc nabízí možnost určitého zjednodušení pole bodů, pokud se opakuje určitá závislost mezi sousedními body. Aby se předešlo milnému popisu v důsledku přerušení hrany kontury objektu, je vyhodnocován konvexní obal dané kontury. Konvexní obaly nalezených kontur objektů jsou popisovány následujícími příznaky:

1. Výška a šířka v ortogonálním systému – funkce *cvBoundingRect(…)*
2. Obvod konvexního obalu – funkce *cvArcLength(…)*
3. Plocha konvexního obalu – funkce *cvContourArea(…)*
4. Minimální poloměr kružnice opsané – funkce *cvMinEnclosingCircle(…)*
5. Jasová hodnota v místě těžiště objektu

Při testování bylo používáno více příznaků pro popis, ale ve finální verzi programu byly tyto příznaky zredukovány na předchozí seznam.

3.3.3.8 Klasifikace objektů

Objekty byly klasifikovány do třech základních tříd:

1. Robot

2. Míč
3. Ostatní objekty

Z pozorování byly stanoveny intervaly jednotlivých příznaků, v kterých se jednotlivé objekty vyskytují. Seznam těchto intervalů je uveden v Tabulce 3.5.

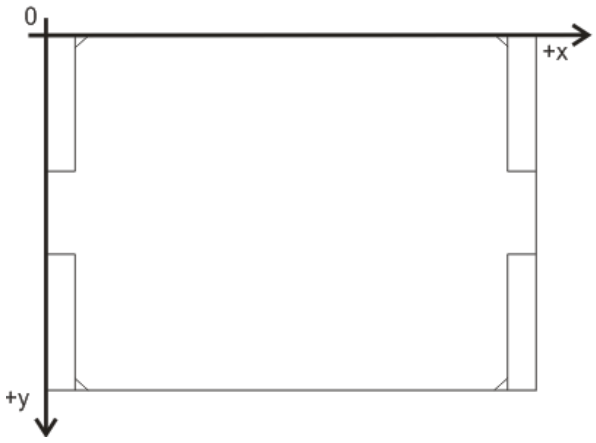
Tabulka 3.5: Hodnoty jednotlivých příznaků

Objekt	Výška	Šířka	Obvod	Plocha	Poloměr	Jas
Robot	> 4	> 4	$\langle 45; 80\rangle$	$\langle 150; 260\rangle$	$\langle 9{,}5; 17\rangle$	> 200
Míč	> 4	> 4	$\langle 25; 35\rangle$	$\langle 65; 90\rangle$	$\langle 5; 6{,}5\rangle$	-

Klasifikátorem byla soustava podmínek, které vycházely z Tabulky 3.5. Aby byl daný objekt do dané třídy klasifikován, musel splňovat všechny výše zmíněně intervaly. Z důvodu toho, že byl popisován objekt, který se nalézal v mírně zkresleném obraze, jsou intervaly v Tabulce 3.5 větší, než by se mohlo u takovéto úlohy zdát.

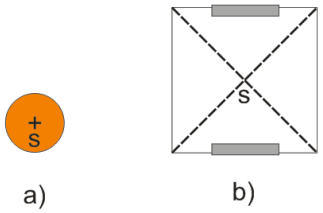
3.3.3.9 Určení pozice objektů

Pro určení pozice objektů, je nutné nejdříve definovat souřadný systém. Systém vychází ze souřadnic nalezeného hřiště. Orientace ortogonálního souřadného systému jsou zobrazeny na Obrázku 3.34.



Obrázek 3.34: Souřadný systém

Při znalosti souřadného systému, je nutné definovat, jakým způsobem se bude určovat pozice jednotlivých objektů. Na Obrázku 3.35 jsou definovány středy jednotlivých objektů, pomocí kterých bude určena jejich poloha.



Obrázek 3.35:Středy jednotlivých objektů: a) Střed míče, b) Střed robotu

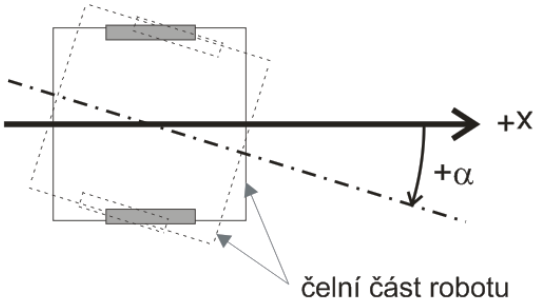
Střed míče se nachází v jeho těžišti, proto určení středu u míče není obtížné. Vypočítat geometrické těžiště lze např. pomocí podílu momentu první a nultého

řádu. Obtížnější situace je u robotu, protože k jeho rozpoznání slouží pravoúhlý trojúhelník. Těžiště pravoúhlého trojúhelníku, který je připevněn na robotu, nemá stejné souřadnice jako střed robotu, proto bylo nutné zvolit jiný způsob určení středu. Ve výsledné verzi programu byl k určení středu robotu použit střed kružnice opsané. Ukázky kružnice opsané u robotu v různých pozicích jsou zobrazeny na Obrázku 3.36, kde je zelenou barvou vyznačena kružnice opsaná. Střed této kružnice opsané je považován za střed robotu. Výsledné souřadnice jsou pomocí vygenerovaných transformačních map metodou nejbližšího souseda přepočítány do nezkresleného obrazu

Obrázek 3.36: Určení středu pomocí kružnice opsané

3.3.3.10 Určení natočení robotů

Tato část se věnuje určení natočení robotu. Nejdříve je nutné stanovit, v jakém smyslu se bude dané natočení měřit. Smysl otáčení je popsán na Obrázku 3.37.



Obrázek 3.37: Smysl natáčení robotu

Algoritmus, pomocí kterého je určeno natočení robotu, lze popsat pomocí následujících kroků:

1. Nejdříve je v binárním obraze hran vyříznut region, v kterém se nalézá zkoumaný robot.
2. Vyříznutý region je vstupním obrazem pro Houghovu transformaci, pomocí které je získán výčet možných přímek, které byly v regionu nalezeny.
3. Následně se z přímek vybere ta, která se svým úhlem natočení nejvíce blíží 45 nebo 135 °.
4. Dáje je nutné přibližně odhadnout, kde se nalézají dominantní body trojúhelníku. Dominantní body jsou takové body, které se podobají

bodům ideálního trojúhelníku. Nejdříve je nutné podle (48) nalézt body, které určují přeponu:

$$[a,b] = \underset{0\le i<n; i<j<n \;\; i,j\in\mathbb{N}}{\operatorname{argmax}} \sqrt{\left(p_x(i)-p_x(j)\right)^2+\left(p_y(i)-p_y(j)\right)^2}, \tag{48}$$

kde $p_x(i)$ souřadnice x i-tého prvku v seznamu všech bodů, které tvoří konturu zkoumaného objektu, a a,b jsou indexy nalezených bodů, které tvoří přeponu.

V dalším kroku je nalezen bod, který je představuje bod „C“ pravoúhlé trojúhelníku. Tento bod je naleze pomocí vztahu (49):

$$c = \underset{0\le i<n \;\; i\in\mathbb{N}}{\operatorname{argmax}} \left(\sqrt{\left(p_x(i)-p_x(a)\right)^2+\left(p_y(i)-p_y(a)\right)^2} + \sqrt{\left(p_x(i)-p_x(b)\right)^2+\left(p_y(i)-p_y(b)\right)^2}\right). \tag{49}$$

Dále je nutné určit, zda přímka, která byla vybrána pomocí Houghovi transformace, je odvěsna či přepona. Nejdříve jsou spočteny vzdálenosti jednotlivých bodů $\{p(a); p(b); p(c)\}$ od vybrané přímky. A poté je vybrána taková kombinace dvou bodů, pro kterou je součet vzdáleností od přímky nejmenší.

5. Podle úhlu, typu přímky (odvěsna, přepona) a podle bodu, který není součástí přímky, je určeno výsledné natočení robotu.

Přesnost určení úhlu natočení

Přesnost úhlu natočení byla určena pro různé světelné podmínky následujícím způsobem:

- Robot opatřený reflexní fólií byl umístěn do středu hřiště.
- Pod robotem byl na černém podkladu šedá stupnice úhlového natočení s dílkem po pěti stupních. Tmavá barva byla zvolena proto, aby při měření docházelo k co nejmenším změnám v prostředí.
- Měření bylo prováděno s krokem 15 °.

Fotografie z měření úhlu natočení je zobrazena na Obrázku 3.38.

Obrázek 3.38: Měření natočení robotu

Tabulka 3.6 zobrazuje statisticky zpracovaná data z měření úhlu natočení pro různé světelné podmínky. Tato tabulka vznikla vyhodnocením 11 tisíců záznamů. V tabulce představuje α_s skutečný úhel natočení, $\bar{\alpha}_m$ průměrnou hodnotu úhlu natočení, $s_\alpha[°]$ výběrovou směrodatnou odchylku měřeného úhlu, E_1 je intenzita osvětlení při jedné zapnuté jedné řadě zářivek, E_2 je intenzita osvětlení při obou řadách rozsvícených zářivek a E_0 je intenzita při všech zářivkách zhasnutých.

Tabulka 3.6: Statistické vyhodnocení měřeného úhlu natočení

	E_2		**E_1**		**E_0**	
$\alpha_s[°]$	$\bar{\alpha}_m[°]$	$s_\alpha[°]$	$\bar{\alpha}_m[°]$	$s_\alpha[°]$	$\bar{\alpha}_m[°]$	$s_\alpha[°]$
0	0,0	0,1	-0,8	1,0	0,1	0,4
15	15,0	0,0	17,0	0,2	15,9	1,6
30	29,5	1,9	31,0	0,0	25,9	1,2
45	41,0	0,1	45,0	0,1	47,1	2,1
60	60,7	1,6	59,9	0,3	58,7	0,8
75	71,9	1,0	75,1	0,5	77,0	0,2
90	89,2	1,8	88,0	0,0	89,0	0,0

Z Tabulky 3.6 je možné pozorovat, že průměrná hodnota natočení se v žádném případě neliší o více jak 4,1 ° a výběrová směrodatná odchylka se nepřesahuje 2,1°. Odchylka v průměrné hodnotě může být také způsobena nedokonalostí zařízení na měření skutečného úhlu. Výběrové směrodatná odchylka je počítána podle vztahu (50) a průměrná hodnota podle vztahu (51).

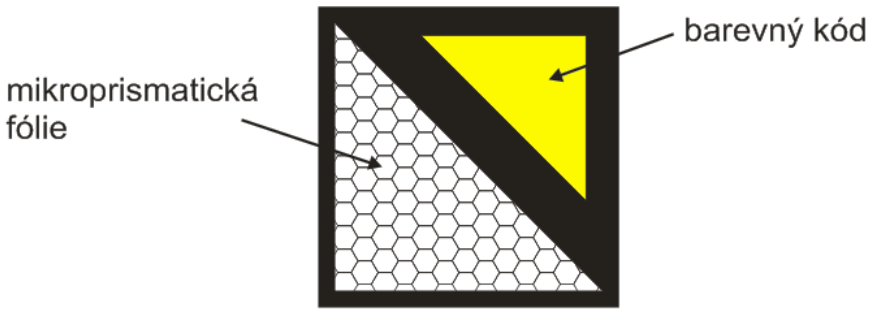
$$\bar{\alpha} = \frac{\sum_{i=1}^{N} \alpha_i}{N} \tag{50}$$

$$s_\alpha = \sqrt{\frac{\sum_{i=1}^{N} (\alpha_i - \bar{\alpha})^2}{N-1}} \tag{51}$$

3.3.3.11 Identifikace robotů

Předchozí podkapitoly se věnovali tomu, jak v obraze nějaký druh objektu najít. Tato podkapitola se zabývá tím, jak od sebe jednotlivé roboty odlišit. Tato část již není v zadání Diplomové práce, ale přesto byla ve zjednodušené formě implementována.

Základem pro rozpoznávání je barevný úsek umístěný na robotu (viz Obrázek 3.38). Tento barevný úsek slouží k rozpoznání jednotlivých robotů od sebe.



Obrázek 3.39: Popis značení robotu

Na rozpoznávání byly použity následující barvy v HSV formátu:

- Žlutá – HSV(60, 100, 90)
- Fialová – HSV(300, 49, 90)
- Světle modrá – HSV(180, 55, 100)

Algoritmus pro identifikaci robotů je možné rozepsat do následujících kroků:

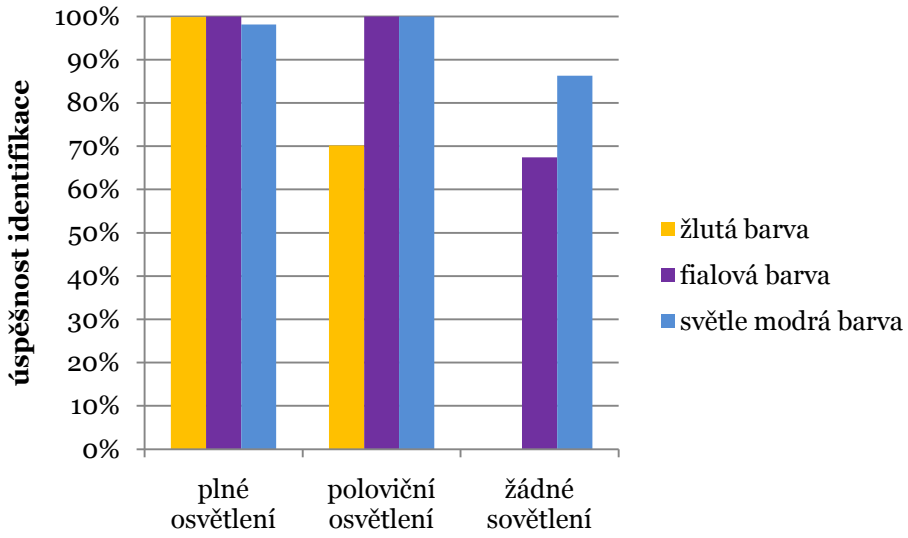
1. Ze středového bodu robotu je vedena úsečka o délce 11 pixelů a natočením $\alpha - 45°$.
2. Pomocí funkce *cvSampleLine(...)* je získán vektor pixelů s barevným formátem RGB.
3. Z důvodu úspory výkonu je pouze tento vektor převeden do formátu HSV.
4. Směrem od středu je hledána skokovitá změna jasu, která určuje počátek barevného kódu.
5. V místech barevného kódu je zprůměrována a vyhodnocena H složka. Podle hodnoty H složky je poté rozhodnuto, o jaký druh robota se jedná.

V Tabulce 3.7 je uvedena úspěšnost identifikace jednotlivých robotů. Pro každou barvu při určitých světelných podmínkách bylo vyhodnocenou minimálně 700 záznamů. Celkem tedy přes 6000 záznamů.

Tabulka 3.7: Úspěšnost identifikace robotů za různých světelných podmínek

barva	úspěšnost – E_2	úspěšnost – E_1	úspěšnost – E_0
žlutá	99,8%	70,2%	0,0%
fialová	100,0%	100,0%	67,4%
světle modrá	98,1%	100,0%	86,3%

Výsledky uvedené v Tabulce 3.7 jsou přehledně zobrazeny pomocí grafu na Obrázku 3.40.



Obrázek 3.40: Graf úspěšnosti identifikace robotů v závislosti na osvětlení a zvolené barvě

Úspěšnost identifikace v Tabulce 3.8 je počítána podle vztahu (52):

$$úspěšnost = \frac{správné\_identifikace}{všechny\_identifikace} \cdot 100\ \% \qquad (52)$$

3.3.4Grafické uživatelské prostředí

Aplikace *„Client_IP“,* která se stará o zpracování obrazu, neumožňuje interakci přes konzolové rozhraní. Proto bylo nutné vyvinout uživatelské prostředí pro ovládání výše zmíněnou aplikaci. Toto prostředí bylo programováno v programu *Micorosoft Visual C++ 2008 – CLR / Windows Forms Application*. Uživatelské prostředí vystupuje jako samostatný proces, který čte a zapisuje data do sdílené paměti.

3.3.4.1 Komunikace s Clinet_IP

Komunikace s procesem, který obstarává zpracování obrazu, je realizována pomocí sdílené paměti. Ve sdílené paměti je uložena struktura, která má následující deklaraci:

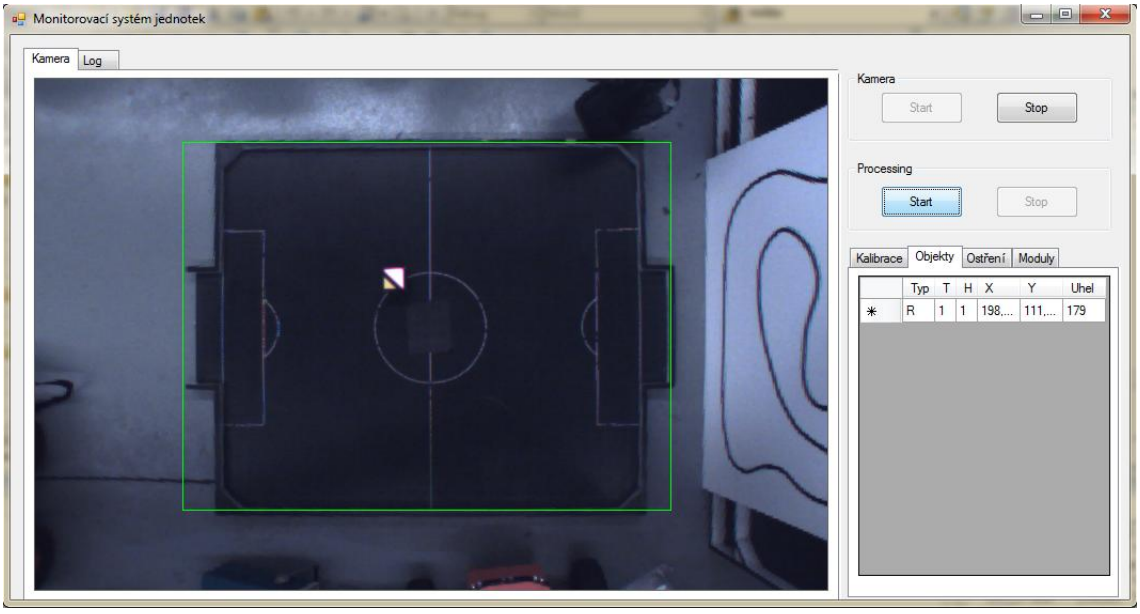
```
typedef struct {
        int                     mod;    // mod ve kterem se processing nachazi
        IplImage                img;    // ukazatel na OpenCV strukturu
        char                    imgBuffer[1071360];  // buffer imgdat
        char                    msg[100];             // zprava
        KALIBRACE               kalibracniParametry;  // parametry kalibrace
        OBJEKT                  objekty[300];         // vlastnosti objektu
        int                     nObjekt;              // pocet nalezenych objektu
        double                  timeStamp;            // casove razitko [s]
        BOOL                    readyToProcess;       // pripraveno vse k processingu ?
} SDILENA_PAMET;
```

Tato i ostatní struktury jsou definovány v hlavičkovém souboru *„komunikace.h“*, ve kterém jsou definovány názvy synchronizačních objektů a některé důležité konstanty. Význam jednotlivých prvků struktury je popsán pomocí následujícího seznamu:

- **mod** – pomocí této proměnné se nastavuje, v jakém módu bude aplikace na zpracování obrazu (kalibrace, ostření, zpracování obrazu nebo pouze přenos obrazu).
- **img** – struktura, která slouží k přenosu informací o obrazu, který je přenášen.
- **imgBuffer** – do tohoto pole se ukládají obrazová data.
- **msg** – pole pro přenos zpráv mezi aplikacemi.
- **kalibracniParametery** – pomocí této struktury se přenáší nastavení pro kalibraci. Struktura slouží také pro přenos hodnot při ostření.
- **objekty** – datové prostor pro nalezené objekty.
- **nObjekt** – počet nalezených objektů.
- **timeStamp** – časové razítko poslední vyhodnocené scény. Obsahuje čas, ve kterém byl snímán obraz kamerou.
- **readyToProces** – proměnná, která informuje uživatelské prostředí o tom, jestli je aplikace na zpracování obrazu připravena ke zpracovávání obrazu.

3.3.4.2 Přenos a zobrazování obrazu

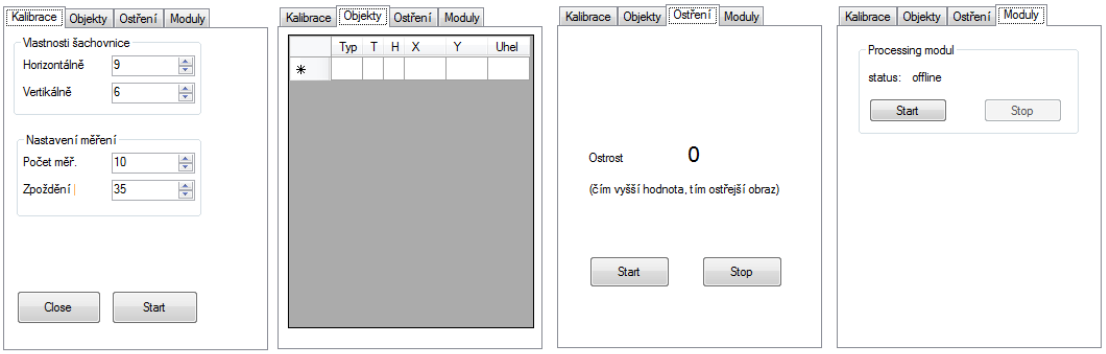
Obrazová data jsou uložena ve sdílené paměti v poli *imgBuffer* a vlastnosti obrazu ve struktuře *img*. K těmto datům je možné přistupovat přes mutex MUTEX_IMAGE_STREAM. K zobrazení obrazových dat je použit ovládací prvek PictureBox. Do tohoto prvku je možné při správném nastavení vykreslovat obrazová data. Vykreslení obstarává funkce *StretchDIBits*. Tato funkce je volána v *Timeru* každých 60 ms. Okno uživatelského prostředí je zobrazeno na Obrázku 3.40.



Obrázek 3.41: Okno uživatelského prostředí

3.3.4.3 Funkce uživatelského prostředí

Pomocí uživatelského prostředí je možné spustit či vypnout přenos obrazu (panel *„Kamera“*). Dále je možné spouštět či vypnout zpracovávání obrazu (panel *„Processing“*). V levé spodní části okna se nalézá panel se záložkami. Jednotlivé záložky jsou zobrazeny na Obrázku 3.42.



Obrázek 3.42: Záložky panelu: a) Kalibrace, b) Objekty, c) Ostření, d) Moduly

Význam jednotlivých záložek je popsán následujícím seznamem:

- **Kalibrace** – záložka slouží k nastavení parametrů kalibrace (vlastnosti šachovnice, počet měření). Stisknutím tlačítka „Start“ se spustí kalibrace kamery.
- **Objekty** – v této záložce se zobrazují aktuálně nalezené objekty ve scéně.
- **Ostření** – pomocí této záložky je možné spustit ostření. V panelu se poté zobrazuje kvantitativní míra ostrosti.
- **Moduly** – zde je možné spouštět, monitorovat a vypínat jednotlivé moduly.

4 ZÁVĚR

V úvodní části této práce byly popsány teoretické poznatky z počítačového zpracování obrazu. Postupně zde byly zmíněny všechny základní kroky zpracování obrazu.

V kapitole (3) je nejdříve popsána scéna, ve které se monitorovací systém nalézá. Měřením byla zjištěna intenzita osvětlení scény při všech zapnutých zářivkách, zapnuté jedné řadě zářivek a všech zářivkách zhasnutých. Dále bylo zjištěno, že v důsledku konstrukce nad hřištěm dochází k nehomogennímu osvětlení hřiště. Aby byla zajištěna co nejvyšší nezávislost na okolním osvětlení, byla na vrchní část robotů umístěna mikroprismatická fólie.

Další část se věnuje digitální kameře, u které bylo zjištěno silné zarušení. Toto zarušení nastávalo pouze při snímkovací frekvenci 30 fps. Při ostatních snímkovacích frekvencích toto silné zarušení nenastávalo. Při nezarušeném obraze bylo provedeno měření, které ukázalo, že nejvíce je šumem zatížená modrá složka obrazu. Naopak červená složka byla zašuměná velmi málo. Pomocí funkcí knihovny *OpenCV* byla odstraněna soudkovitost objektivu. Přetransformovaný obraz byl poměrně kvalitní, ale obsahoval nespojitosti, které vedly k nekvalitní hranové reprezentaci. Proto na zpracování obrazu byl použit zkreslený obraz.

Předzpracování obrazu bylo omezeno na filtraci pomocí filtru s konvoluční maskou s Gaussovým rozložením. Segmentace probíhala pomocí hranové reprezentace, která byla získána pomocí Cannyho hranového detektoru. Z hranové reprezentace byly vybrány jednotlivé objekty, ke kterým byly spočítány příznaky, podle kterých probíhala klasifikace. U nalezených robotů, bylo určeno natočení pomocí Houghovy transformace. Výběrová směrodatná odchylka měření natočení nepřesáhla 2,1 ° a průměrná hodnota natočení se od skučených hodnot lišila maximálně o 4,5 °. Tato odchylka mohla být způsobena i nedokonalým měřením skutečného natočení. Dále byla testována identifikace jednotlivých robotů. Měření byly prováděny pro tři barvy (žlutá, fialová a světle modrá). Při plném osvětlení nebyla úspěšnost identifikace u všech barev nižší než 98%. Při polovičním osvětlení byl algoritmus úspěšný u žluté barvy na 70,1% a u ostatních barev byl stoprocentní. Jeden celý cyklus, který zahrnuje předzpracování, segmentaci, popis, klasifikaci, měření pozice a identifikaci, trval v průměru 11 ms pro šest robotů a jeden míč na scéně.

Aplikace byla navržena tak, aby bylo možné sdílet důležitá data mezi více aplikacemi. Nyní se jakákoliv aplikace může připojit na sdílenou paměť a vyčítat informace o poloze objektů nebo zpracovávat obrazová data. Přes sdílenou paměť je také možné aplikaci, zpracovávající obrazová data, řídit. Pro účely

řízení bylo naprogramováno uživatelské prostředí, které umožňuje aplikaci *„Client_IP.exe“* ovládat.

Aplikaci, která zpracovává obraz, by bylo možné vylepšit pomocí rozhodovacího stromu, který by mohl lépe klasifikovat jednotlivé objekty. Dále by bylo možné vylepšit způsob identifikace, tak aby bylo možné rozpoznávat větší množství robotů.

Literatura

[1] HORÁK, Karel, et al. *Počítačové vidění*. Brno : Skriptum VUT, 2008. 132 s.

[2] HLAVÁČ, Václav; ŠONKA, Milan. *Počítačové vidění*. Praha : Grada, 1992. 252 s. ISBN 80-85424-67-3.

[3] BRADSKI, Gary; KAEHLER, Adrian. *Learning OpenCV*. Cambrige : O'Reilly, 2008. 571 s. ISBN 978-0-596-51613-0.

[4] *Čočka (optika)*. In Wikipedia : the free encyclopedia [online]. St. Petersburg (Florida) : Wikipedia Foundation, 12.2.2007, last modified on 20.4.2011 [cit. 2011-05-19]. Dostupné z WWW: <http://cs.wikipedia.org/wiki/%C4%8Co%C4%8Dka_(optika)>.

[5] JANOUCH, Michal. *Zpracování obrazu jednočipovým mikroprocesorem*. Praha, 2008. 86 s. Diplomová práce. České vysoké učení technické v Praze, Fakulta elektrotechnická. Dostupné z WWW: <http://imr.felk.cvut.cz/robotour/files/dp_michal_janouch/DP-Michal_Janouch.pdf>.

[6] HLAVÁČ, Václav. *Jasové a goemetrické trnsformace* [online]. Praha: Fakulta elektrotechnická, ČVUT, [cit. 2011-05-11]. Dostupný z WWW: <http://cmp.felk.cvut.cz/~hlavac/TeachPresCz/11DigZprObr/18BrightGeomTxCz.pdf>

[7] JAN, Jiří. *Analýza obrazů (5)* [online]. Brno: Fakulta elektrotechnická, VUT, [cit. 2011-05-11]. Dostupný z WWW: <https://www.vutbr.cz/elearning/file.php/98792/Prednasky_new/MASO_9_new.pdf>

[8] WALEK, Petr. *Analýza signálů a obrazů - počítačové cvičení č.5*.[online]. Brno: Fakulta elektrotechnická, VUT, [cit. 2011-05-11]. Dostupný z WWW: <https://www.vutbr.cz/elearning/mod/resource/view.php?id=113985>

[9] KEKRT, Daniel, at al. *Dvoudimenzionální interpolacní metody*[online]. Praha: Katedra radioelektroniky, ČVUT.

[10] WOLF, Joerg. *The Secrets of Robot Football* [online]. Plymouth: University of Plymouth, [cit. 2011-05-11]. Dostupný z WWW: <http://www.swrtec.de/swrtec/research/publications/Secrets_of_Robot Football.pdf>

Seznam zkratek

CCD	Charge-Coupled device
CMOS	Complementary Metal–Oxide–Semiconductor
HSV	Hue, Saturation, Value
IBL	Instance-Based Learning algorithms
LED	Light-Emitting Diode
PC	Personal Computer
RGB	Red, Green, Blue
SMD	Surface Mount Device
USB	Universal Serial Bus

Seznam příloh

Příloha č. 1: Konfigurace notebooku

Parametr	Hodnota
Procesor	Intel Core 2 Duo T5870
Frekvence procesoru	2,0 GHz
Druh operační paměti	DDR2, 800 MHz
Velikost operační paměti	4 GB
Velikost pevného disku	500 GB